

Springer

Handbook *of* Robotics



Siciliano
Khatib
Editors

Springer Handbook of Robotics

Springer Handbooks provide a concise compilation of approved key information on methods of research, general principles, and functional relationships in physical sciences and engineering. The world's leading experts in the fields of physics and engineering will be assigned by one or several renowned editors to write the chapters comprising each volume. The content is selected by these experts from Springer sources (books, journals, online content) and other systematic and approved recent publications of physical and technical information.

The volumes are designed to be useful as readable desk reference books to give a fast and comprehensive overview and easy retrieval of essential reliable key information, including tables, graphs, and bibliographies. References to extensive sources are provided.

Springer Handbook of Robotics

Bruno Siciliano, Oussama Khatib (Eds.)

With DVD-ROM, 953 Figures, 422 in four color and 84 Tables



Springer

Editors:

Professor Bruno Siciliano
PRISMA Lab
Dipartimento di Informatica e Sistemistica
Università degli Studi di Napoli Federico II
Via Claudio 21, 80125 Napoli, Italy
siciliano@unina.it

Professor Oussama Khatib
Artificial Intelligence Laboratory
Department of Computer Science
Stanford University
Stanford, CA 94305-9010, USA
khatib@cs.stanford.edu

Library of Congress Control Number: 2007942155

ISBN: 978-3-540-23957-4 e-ISBN: 978-3-540-30301-5

This work is subject to copyright. All rights reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilm or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September, 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media
springer.com

© Springer-Verlag Berlin Heidelberg 2008

The use of designations, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Product liability: The publisher cannot guarantee the accuracy of any information about dosage and application contained in this book. In every individual case the user must check such information by consulting the relevant literature.

Production and typesetting: le-tex publishing services oHG, Leipzig
Senior Manager Springer Handbook: Dr. W. Skolaut, Heidelberg
Typography and layout: schreiberVIS, Seeheim
Illustrations: Hippmann GbR, Schwarzenbruck
Cover design: eStudio Calamar Steinen, Barcelona
Cover production: WMXDesign GmbH, Heidelberg
Printing and binding: Stürtz GmbH, Würzburg

Printed on acid free paper

SPIN 10918401 89/3180/YL 5 4 3 2 1 0

Foreword

My first introduction to robotics came via a phone call in 1964. The caller was Fred Terman, the author of the world-famous *Radio Engineer's Handbook*, who was at the time Provost of Stanford University. Dr. Terman informed me that a computer science professor, John McCarthy, had just been awarded a large research grant, part of which required the development of computer-controlled manipulators. Someone had suggested to Terman that it would be prudent if the mathematically oriented McCarthy had some contact with mechanical designers. Since I was the only one on the Stanford faculty whose specialty was mechanism design, Terman decided to phone me, even though we had never met and I was a young assistant professor fresh out of graduate school with only 2 years at Stanford.

Dr. Terman's phone call led me to a close association with John McCarthy and the Stanford Artificial Intelligence Laboratory ([SAIL](#)) that he founded. Robotics became one of the pillars of my entire academic career, and I have maintained my interest in teaching and researching the subject through to the present day.

The modern history of robotic manipulation dates from the late 1940s when servoed arms were developed in connection with master-slave manipulator systems used to protect technicians handling nuclear materials. Developments in this area have continued to the present day. However, in the early 1960s there was very little academic or commercial activity in robotics. The first academic activity was the thesis of H. A. Ernst, in 1961, at MIT. He used a slave arm equipped with touch sensors, and ran it under computer control. The idea in his study was to use the information from the touch sensors to guide the arm.

This was followed by the [SAIL](#) project and a similar project started by Professor Marvin Minsky at MIT, which were the only sizeable academic ventures into robotics at that time. There were a few attempts at commercial manipulators, primarily in connection with part production in the automotive industry. In the USA there were two different manipulator designs that were being experimented with in the auto industry; one came from American Machine and Foundry (AMF) and the other from Unimation, Inc.

There were also a few mechanical devices developed as hand, leg, and arm prosthetics, and, a bit later, some

exoskeletal devices to enhance human performance. In those days there were no microprocessors. So, these devices were either without computer control, or tethered to a remote so-called minicomputer, or even a mainframe computer.

Initially, some in the computer science community felt that computers were powerful enough to control any mechanical device and make it perform satisfactorily. We quickly learned that this was not to be the case. We started on a twofold track. One was to develop particular devices for [SAIL](#), so that hardware demonstrations and proof-of-concept systems were available for the fledgling robotics community to experiment with. The other track, which was more or less moonlighted from the work at [SAIL](#), was the development of a basic mechanical science of robotics. I had a strong feeling that a meaningful science could be developed, and that it would be best to think in terms of general concepts rather than concentrate exclusively on particular devices.

Fortuitously, it turned out that the two tracks supported each other very naturally and, most importantly, the right students were interested in doing their research in this area. Hardware developments proved to be specific examples of more general concepts, and the students were able to develop both the hardware and the theory.

Originally, we purchased an arm in order to get started quickly. A group at Rancho Los Amigos Hospital, in Los Angeles, was selling a tongue-switch-controlled motor-driven exoskeleton arm to assist patients without muscular control of their arms. We purchased one of these, and connected it to a time-shared PDP-6 computer. The device was named *Butterfingers*; it was our first experimental robot. Several films demonstrating visual feedback control, block stacking tasks, and obstacle avoidance were made with *Butterfingers* as the star performer.

The first manipulator that we designed on our own was known simply as the *Hydraulic Arm*. As its name implies, it was powered by hydraulics. The idea was



Bernard Roth
Professor of Mechanical
Engineering
Stanford University

to build a very fast arm. We designed special rotary actuators, and the arm worked well. It became the experimental platform for testing the first ever dynamic analysis and time-optimal control of a robotic arm. However, its use was limited since the design speeds were much faster than required due to the limitations of the computational, planning, and sensing capabilities that were common at that time.

We made an attempt to develop a truly digital arm. This led to a snake-like structure named the *Orm* (the Norwegian word for snake.) The *Orm* had several stages, each with an array of inflatable pneumatic actuators that were either fully extended or fully contracted. The basic idea was that, even though only a finite number of positions in the workspace could be reached, these would be sufficient if there were a large number of positions. A small prototype proof-of-concept *Orm* was developed. It led to the realization that this type of arm would not really serve the SAIL community.

The first truly functional arm from our group was designed by Victor Scheinman, who was a graduate student at the time. It was the very successful *Stanford Arm*, of which over ten copies were made as research tools to be used in various university, government, and industrial laboratories. The arm had six independently driven joints; all driven by computer-controlled servoed, DC electric motors. One joint was telescoping (prismatic) and the other five were rotary (revolute).

Whereas the geometry of *Butterfingers* required an iterative solution of the inverse kinematics, the geometric configuration of the *Stanford Arm* was chosen so that the inverse kinematics could be programmed in any easy-to-use time-efficient closed form. Furthermore, the mechanical design was specifically made to be compatible with the limitations inherent in timeshare computer control. Various end-effectors could be attached to act as hands. On our version, the hand was in the form of a vise-grip jaw, with two sliding fingers driven by a servoed actuator (hence, a true seventh degree of freedom). It also had a specially designed six-axis wrist force sensor. Victor Scheinman went on to develop other important robots: the first was a small humanoid arm with six revolute joints. The original design was paid for by Marvin Minsky at the MIT AI Lab. Scheinman founded Vicarm, a small company, and produced copies of this arm and the *Stanford Arm* for other labs. Vicarm later became the West Coast Division of Unimation, Inc., where Scheinman designed the PUMA manipulator under General Motors sponsorship through Unimation. Later, for a company called Automatrix, Scheinman developed the novel *Robot World* multirobot system. After Scheinman

left Unimation, his colleagues Brian Carlisle and Bruce Shimano reorganized Unimation's West Coast Division into Adept, Inc., which to this day is the largest US manufacturer of assembly robots.

Quickly, the modern trend of carefully detailed mechanical and electronic design, optimized software, and complete system integration became the norm; to this day, this combination represents the hallmark of most highly regarded robotic devices. This is the basic concept behind *mechatronic*, a word coined in Japan as a concatenation of the words mechanics and electronics. Mechatronics that relies on computation is the essence of the technology inherent in robotics as we know it today.

As robotics developed around the world, a large number of people started working on various aspects, and specific subspecialties developed. The first big division was between people working on manipulators and those working on vision systems. Early on, vision systems seemed to hold more promise than any other method for giving robots information about their environment.

The idea was to have a television camera capture pictures of objects in the environment, and then use algorithms that allowed the computer images of the pictures to be analyzed, so as to infer required information about location, orientation, and other properties of objects. The initial successes with image systems were in problems dealing with positioning blocks, solving object manipulation problems, and reading assembly drawings. It was felt that vision held potential for use in robotic systems in connection with factory automation and space exploration. This led to research into software that would allow vision systems to recognize machine parts (particularly partially occluded parts, as occurred in the so-called "bin-picking" problems) and ragged-shaped rocks.

After the ability to "see" and move objects became established, the next logical need had to do with planning a sequence of events to accomplish a complex task. This led to the development of planning as an important branch in robotics. Making fixed plans for a known fixed environment is relatively straightforward. However, in robotics, one of the challenges is to let the robot discover its environment, and to modify its actions when the environment changes unexpectedly due to errors or unplanned events. Some early landmark studies in this area were carried out using a vehicle named *Shakey*, which, starting in 1966, was developed by Charlie Rosen's group at the Stanford Research Institute (now called SRI). *Shakey* had a TV camera, a trian-

gulating range finder, bump sensors, and was connected to DEC PDP-10 and PDP-15 computers via radio and video links.

Shakey was the first mobile robot to reason about its actions. It used programs that gave it the ability for independent perception, world modeling, and action generation. Low-level action routines took care of simple moving, turning, and route planning. Intermediate-level actions combined the low-level ones in ways that accomplished more complex tasks. The highest level programs could make and execute plans to achieve high-level goals supplied by a user.

Vision is very useful for navigation, locating objects, and determining their relative positions and orientation. However, it is usually not sufficient for assembling parts or working with robots where there are environmental constraining forces. This led to the need to measure the forces and torques generated by the environment, on a robot, and to use these measurements to control the robot's actions. For many years, force-controlled manipulation became one of the main topics of study at [SAIL](#), and several other labs around the world. The use of force control in industrial practice has always lagged the research developments in this area. This seems to be due to the fact that, while a high level of force control is very useful for general manipulation issues, specific problems in very restricted industrial environments can often be handled with limited, or no, force control.

In the 1970s, specialized areas of study such as walking machines, hands, automated vehicles, sensor integration, and design for hostile environments began to develop rapidly. Today there are a large number of different specialties studied under the heading of robotics. Some of these specialties are classical engineering subject areas within which results have been developed that have been particularized to the types of machines called robots. Examples here are kinematics, dynamics, controls, machine design, topology, and trajectory planning. Each of these subjects has a long history predating the study of robotics; yet each has been an area of in-depth robotics research in order to develop its special character in regard to robotic-type systems and applications. In doing this specialized development, researchers have enriched the classical subjects by increasing both their content and scope.

At the same time that the theory was being developed, there was a parallel, although somewhat separate, growth of industrial robotics. Strong commercial development occurred in Japan and Europe, and there was also continued growth in the USA. Industrial associa-

tions were formed (the Japan Robot Association was formed in March 1971, and the Robotic Industries Association (RIA) was founded in 1974 in the USA) and trade shows, together with application-oriented technical sessions, were introduced and held on a regular basis. The most important were the International Symposium on Industrial Robots, the Conference on Industrial Robot Technology (now called the International Conference on Industrial Robot Technology), and the RIA annual trade show, which is now called the International Robots and Vision Show and Conference.

The first regular series of conferences emphasizing research, rather than the industrial, aspects of robotics, was inaugurated in 1973. It was sponsored jointly by the International Center for Mechanical Sciences (CISM), based in Udine, Italy, and the International Federation for the Theory of Mechanisms and Machines (IFTToMM). (Although IFTToMM is still used, its meaning has been changed to the International Federation for the Promotion of Mechanism and Machine Science.) It was named the Symposium on Theory and Practice of Robots and Manipulators (RoManSy). Its trademark was an emphasis on the mechanical sciences and the active participation of researchers from Eastern and Western Europe as well as North America and Japan. It is still held biannually. On a personal note, it is at RoManSy where I first met each of the editors of this Handbook: Dr. Khatib in 1978 and Dr. Siciliano in 1984. They were both students: Bruno Siciliano had been working on his PhD for about one year, and Oussama Khatib had just completed his PhD research. In both cases, it was love at first sight!

RoManSy was quickly joined by a host of other new conferences and workshops; today there are a large number of research oriented robotics meetings that take place through the year in many countries. Currently, the largest conference is the International Conference on Robotics and Automation (ICRA), which regularly draws well over 1000 participants.

In the beginning of the 1980s, the first real textbook on robotic manipulation in the USA was written by Richard "Lou" Paul (Richard P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*, The MIT Press, Cambridge, MA, 1981). It used the idea of taking classical subjects in mechanics and applying them to robotics. In addition there were several topics developed directly from his thesis research at [SAIL](#). (In the book, many examples are based on Scheinman's Stanford Arm.) Paul's book was a landmark event in the USA; it created a pattern for several influential future textbooks and also encouraged the creation of

specialized robotics courses at a host of colleges and universities.

At about this same time, new journals were created to deal primarily with research papers in the areas related to robotics. The *International Journal of Robotics Research* was founded in the spring of 1982, and three years later the *IEEE Journal of Robotics and Automation* (now the *IEEE Transactions on Robotics*) was founded.

As microprocessors became ubiquitous, the question of what is or is not a robot came more into play. This issue has, in my mind, never been successfully resolved. I do not think a definition will ever be universally agreed upon. There are of course the science fiction creatures-from-outer-space varieties, and the robots of the theater, literature, and the movies. There are examples of imaginary robot-like beings that predate the industrial revolution, but how about more down-to-Earth robots? In my view the definition is essentially a moving target that changes its character with technological progress. For example, when it was first developed, a ship's gyro auto-compass was considered a robot. Today, it is not generally included when we list the robots in our world. It has been demoted and is now considered an automatic control device.

For many, the idea of a robot includes the concept of multifunctionality, meaning the device is designed and built with the ability to be easily adapted or reprogrammed to do different tasks. In theory this idea is valid, but in practice it turns out that most robotic devices are multifunctional in only a very limited arena. In industry it was quickly discovered that a specialized machine, in general, performs much better than a general purpose machine. Furthermore, when the volume of production is high enough, a specialized machine can cost less to manufacture than a generalized one. So, specialized robots were developed for painting, riveting, quasiplanar parts assembly, press loading, circuit board stuffing, etc. In some cases robots are used in such specialized ways that it becomes difficult to draw the line between a so-called robot and an adjustable piece of "fixed" automation. Much of this practical unfolding is contrary to the dream of the pioneers in robotics, who had hoped for the development of general purpose machines that would do "everything", and hence sell in great enough volume to be relatively inexpensive.

My view is that the notion of a robot has to do with which activities are, at a given time, associated with people and which are associated with machines. If a machine suddenly becomes able to do what we normally associate with people, the machine can be upgraded in classification and classified as a robot. After a while,

people get used to the activity being done by machines, and the devices get downgraded from "robot" to "machine". Machines that do not have fixed bases, and those that have arm- or leg-like appendages have the advantage of being more likely called robots, but it is hard to think of a consistent set of criteria that fits all the current naming conventions.

In actuality any machines, including familiar household appliances, which have microprocessors directing their actions can be considered as robots. In addition to vacuum cleaners, there are washing machines, refrigerators, and dishwashers that could be easily marketed as robotic devices. There are of course a wide range of possibilities, including those machines that have sensory environmental feedback and decision-making capabilities. In actual practice, in devices considered to be robotic, the amount of sensory and decision making capability may vary from a great deal to none.

In recent decades the study of robotics has expanded from a discipline centered on the study of mechatronic devices to a much broader interdisciplinary subject. An example of this is the area called human-centered robotics. Here one deals with the interactions between humans and intelligent machines. This is a growing area where the study of the interactions between robots and humans has enlisted expertise from outside the classical robotics domain. Concepts such as emotions in both robots and people are being studied, and older areas such as human physiology and biology are being incorporated into the mainstream of robotics research. These activities enrich the field of robotics, as they introduce new engineering and science dimensions into the research discourse.

Originally, the nascent robotics community was focused on getting things to work. Many early devices were remarkable in that they worked at all, and little notice was taken of their limited performance. Today, we have sophisticated, reliable devices as part of the modern array of robotic systems. This progress is the result of the work of thousands of people throughout the world. A lot of this work took place in universities, government research laboratories, and companies. It is a tribute to the worldwide engineering and scientific community that it has been able to create the vast amount of information that is contained in the 64 chapters of this Handbook. Clearly these results did not arise by any central planning or by an overall orderly scheme. So the editors of this handbook were faced with the difficult task of organizing the material into a logical and coherent whole.

The editors have accomplished this by organizing the contributions into a three-layer structure. The first layer deals with the *foundations* of the subject. This layer consists of a single part of nine chapters in which the authors lay out the root subjects: kinematics, dynamics, control, mechanisms, architecture, programming, reasoning, and sensing. These are the basic technological building blocks for robotics study and development.

The second layer has four parts. The first of these deals with *robot structures*; these are the arms, legs, hands, and other parts that most robots are made up of. At first blush, the hardware of legs, arms, and hands may look quite different from each other, yet they share a common set of attributes that allows them to all be treated with the same, or closely related, aspects of the fundamentals described in the first layer.

The second part of this layer deals with *sensing and perception*, which are basic abilities any truly autonomous robotic system must have. As was pointed out earlier, in practice, many so-called robotic devices have little of these abilities, but clearly the more advanced robots cannot exist without them, and the trend is very much toward incorporating such capabilities into robotic devices. The third part of this layer treats the subject areas associated with the technology of *manipulation and the interfacing of devices*. The fourth part of this layer is made up of eight chapters that treat *mobile robots and various forms of distributed robotics*.

The third layer consists of two separate parts (a total of 22 chapters) that deal with advanced applications at the forefront of today's research and development. There are two parts to this layer; one deals with *field and service robots*, and the other deals with *human-centered and lifelike robots*. To the uninitiated observer, these chapters are what advanced robotics is all about. However, it is important to realize that many of these extraordinary accomplishments would probably not exist without the previous developments introduced in the first two layers of this Handbook.

It is this intimate connection between theory and practice that has nurtured the growth of robotics and become a hallmark of modern robotics. These two complementary aspects have been a source of great personal satisfaction to those of us who have had the opportunity to both research and develop robotic devices. The contents of this Handbook admirably reflect this complementary aspect of the subject, and present a very useful bringing together of the vast accomplishments which have taken place in the last 50 years. Certainly, the contents of this Handbook will serve as a valuable tool and guide to those who will produce the even more capable and diverse next generations of robotic devices. The editors and authors have my congratulations and admiration.

Stanford, August 2007

Bernard Roth

Foreword



Georges Giralt

Emeritus Research Director
LAAS-CNRS Toulouse

To open this Handbook and unfold the richness of its 64 chapters, we here attempt a brief personal overview to sketch the evolution of robotics in its many aspects, concepts, trends, and central issues.

The modern story of Robotics began about half a century ago with developments in two different directions.

First, let us acknowledge the domain of mechanical arms, ranging from teleoperated tasks on radiation-contaminated products to industrial arms, with the landmark machine UNIMATE – standing for uni(versal)mate. The industrial development of products, mostly around the six-degree-of-freedom serial links paradigm and active research and development, associating mechanical engineering to the control specialism, was the main driving force here. Of particular note nowadays is the successfully pursued effort to design novel application-optimized structures, using powerful sophisticated mathematical tools. In a similar way, an important issue concerns the design and the actual building of arms and hands in the context of human-friendly robots for tomorrow's cognitive robot.

Second, and less well recognized, we should acknowledge the stream of work concerned with themes in artificial intelligence. A landmark project in this area was the mobile robot *Shakey* developed at Stanford International. This work, which aimed to bring together computer science, artificial intelligence, and applied mathematics to develop intelligent machines, remained a secondary area for quite some time. During the 1980s, building strength from many study cases encompassing a spectacular spectrum ranging from rovers for extreme environments (planet exploration, Antarctica, etc.), to service robots (hospitals, museum guides, etc.), a broad research domain arose in which machines could claim the status of intelligent robots.

Hence robotics researches could bring together these two different branches, with intelligent robots categorized in a solely computational way as bounded

rationality machines, expanding on the 1980s third-generation robot definition:

“(robot) . . . operating in the three-dimensional world as a machine endowed with the capacity to interpret and to reason about a task and about its execution, by intelligently relating perception to action.”

The field of autonomous robots, a widely recognized test-bed, has recently benefited from salient contributions in robot planning using the results of algorithmic geometry as well as of a stochastic framework approach applied both to environmental modeling and robot localization problems (SLAM, simultaneous localization and modeling), and further from the development of decisional procedures via Bayesian estimation and decision approaches.

For the last decade of the millennium, robotics largely dealt with the intelligent robot paradigm, blending together robots and machine-intelligence generic research within themes covering advanced sensing and perception, task reasoning and planning, operational and decisional autonomy, functional integration architectures, intelligent human-machine interfaces, safety, and dependability.

The second branch, for years referred to as non-manufacturing robotics, concerns a wide spectrum of research-driven real-world cases pertaining to field, service, assistive, and, later, personal robotics. Here, machine intelligence is, in its various themes, the central research direction, enabling the robot to act:

1. as a human surrogate, in particular for intervention tasks in remote and/or hostile environments
2. in close interaction with humans and operating in human environments in all applications encompassed by human-friendly robotics, also referred to as human-centered robotics
3. in tight synergy with the user, expanding from mechanical exoskeleton assistance, surgery, health care, and rehabilitation into human augmentation.

Consequently, at the turn of the millennium, robotics appears as a broad spectrum of research themes both supporting market products for well-engineered industrial workplaces, and a large number of domain-oriented application cases operating in hazardous and/or

harsh environments (underwater robotics, rough-terrain rovers, health/rehabilitation care robotics, etc.) where robots exhibit meaningful levels of shared autonomy.

The evolution levels for robotics stress the role of theoretical aspects, moving from application domains to the technical and scientific area. The organization of this Handbook illustrates very well these different levels. Furthermore, it rightly considers, besides a body of software systems, front-line matters on physical appearance and novel appendages, including legs, arms, and hands design in the context of human-friendly robots for tomorrow's cognitive robot.

Forefront robotics in the first decade of the current millennium is making outstanding progress, compounding the strength of two general directions:

- short/mid-term application-oriented study cases
- mid/long-term generic situated research

For completeness, we should mention the large number of peripheral, robotics-inspired subjects, quite often concerning entertainment, advertising, and sophisticated toys.

The salient field of human-friendly robotics encompasses several front-line application domains where the robots operate in a human environment and in close interaction with humans (entertainment and education, public-oriented services, assistive and personal robots, etc.), which introduces the critical issue of human-robot interaction.

Right at the core of the field, emerges the forefront topic of personal robots for which three general characteristics should be emphasized:

1. They may be operated by a nonprofessional user;
2. They may be designed to share high-level decision making with the human user;
3. They may include a link to environment devices and machine appendages, remote systems, and operators; the shared decisional autonomy concept (co-autonomy) implied here unfolds into a large set of cutting-edge research issues and ethical problems.

The concept of the personal robot, expanding to robot assistant and universal companion, is a truly great challenge for robotics as a scientific and technical field,

offering the mid/long-term perspective of achieving a paramount societal and economical impact. This introduces, and questions, front-line topics encompassing cognitive aspects: user-tunable human-machine intelligent interfaces, perception (scene analysis, category identification), open-ended learning (understanding the universe of action), skills acquisition, extensive robot-world data processing, decisional autonomy, and dependability (safety, reliability, communication, and operating robustness).

There is an obvious synergistic effort between the two aforementioned approaches, in spite of the necessary framework time differences. The scientific link not only brings together the problems and obtained results but also creates a synergistic exchange between the two sides and the benefits of technological progress.

Indeed, the corresponding research trends and application developments are supported by an explosive evolution of enabling technologies: computer processing power, telecommunications, networking, sensing devices, knowledge retrieval, new materials, micro- and nanotechnologies.

Today, looking to the mid- and long-term future, we are faced with very positive issues and perspectives but also having to respond to critical comments and looming dangers for machines that are in physical contact with the user and may also be capable of unwanted, unsafe behavior. Therefore, there is a clear need to include at the research level safety issues and the topic of multifaced dependability and the corresponding system constraints.

The *Handbook of Robotics* is an ambitious and timely endeavor. It summarizes a large number of problems, questions, and facets considered by 164 authors in 64 chapters. As such it not only provides an efficient display of basic topics and results obtained by researches around the world, but furthermore gives access to this variety of viewpoints and approaches to everyone. This is indeed an important tool for progress but, much more, is the central factor that will establish the two first decades of this millennium as the dawn of robotics, lifted to a scientific discipline at the core of machine intelligence.

Toulouse, December 2007

Georges Giralt

Foreword



Hirochika Inoue

Professor Emeritus
The University of Tokyo

The field of robotics was born in the middle of the last century when emerging computers were altering every field of science and engineering. Having gone through fast yet steady growth via a procession of stages from infancy, childhood, and adolescence to adulthood, robotics is now mature and is expected to enhance the quality of people's lives in society in the future.

In its infancy, the core of robotics consisted of pattern recognition, automatic control, and artificial intelligence. Taking on these

new challenge, scientists and engineers in these fields gathered to investigate novel robotic sensors and actuators, planning and programming algorithms, and architectures to connect these components intelligently. In so doing, they created artifacts that could interact with humans in the real world. An integration of these early robotics studies yielded *hand-eye systems*, the test-bed of artificial intelligence research.

The playground for childhood robotics was the factory floor. Industrial robots were invented and introduced into the factory for automating spraying, spot welding, grinding, materials handling, and parts assembly. Machines with sensors and memories made the factory floor smarter, and its operations more flexible, reliable, and precise. Such robotic automation freed humans from heavy and tedious labor. The automobile, electric appliance, and semiconductor industries rapidly retooled their manufacturing lines into robot-integrated systems. In the late 1970s, the word *mechatronics*, originally coined by the Japanese, defined a new concept of machinery, one in which electronics was fused with mechanical systems, making a wide range of industrial products simpler, more functional, programmable, and intelligent. Robotics and mechatronics exerted an evolutionary impact on the design and operation of manufacturing processes as well as on manufactured products.

As robotics entered its adolescence, researchers were ambitious to explore new horizons. Kinematics, dynamics, and control system theory were refined and applied to real complex robot mechanisms. To plan and carry

out real tasks, robots had to be made cognizant of their surroundings. Vision, the primary channel for external sensing, was exploited as the most general, effective, and efficient means for robots to understand their external situation. Advanced algorithms and powerful devices were developed to improve the speed and robustness of robot vision systems. Tactile and force sensing systems also needed to be developed for robots to manipulate objects. Studies on modeling, planning, knowledge, reasoning, and memorization expanded their intelligent properties. Robotics became defined as the study of intelligent connection of sensing to actuation. This definition covered all aspects of robotics: three scientific cores and one synthetic approach to integrate them. Indeed, system integration became a key aspect of robotic engineering as it allows the creation of lifelike machines. The fun of creating such robots attracted many students to the robotics field.

In advancing robotics further, scientific interest was directed at understanding humans. Comparative studies of humans and robots led to new approaches in scientific modeling of human functions. Cognitive robotics, lifelike behavior, biologically inspired robots, and a psychophysiological approach to robotic machines culminated in expanding the horizons of robotic potential. Generally speaking, an immature field is sparse in scientific understanding. Robotics in the 1980s and 1990s was in such a youthful stage, attracting a great many inquisitive researchers to this new frontier. Their continuous explorations into new realms form the rich scientific contents of this comprehensive volume.

Further challenges, along with expertise acquired on the cutting edge of robotics, opened the way to real-world applications for mature robotics. The early-stage playground gave way to a workshop for industrial robotics. Medical robotics, robot surgery, and in vivo imaging save patients from pain while providing doctors with powerful tools for conducting operations. New robots in such areas as rehabilitation, health care, and welfare are expected to improve quality of life in an aging society. It is the destiny of robots to go everywhere, in the air, under water, and into space. They are expected to work hand in hand with humans in such areas as agriculture, forestry, mining, construction, and hazardous environments and rescue operations, and to find

utility both in domestic work and in providing services in shops, stores, restaurants, and hospitals. In a myriad of ways, robotic devices are expected to support our daily lives. At this point, however, robot applications are largely limited to structured environments, where they are separated from humans for safety sake. In the next stage, their environment will be expanded to an unstructured world, one in which humans, as service takers, will always live and work beside robots. Improved sensing, more intelligence, enhanced safety, and better human understanding will be needed to prepare robots to function in such an environment. Not only technical but also social matters must be considered in finding solutions to issues impeding this progress.

Since my initial research to make a robot turn a crank, four decades have passed. I feel both lucky and happy to have witnessed the growth of robotics from its early beginnings. To give birth to robotics, fundamental technologies were imported from other disciplines. Neither textbooks nor handbooks were available. To reach the present stage, a great many scientists and engineers have challenged new frontiers; advancing robotics, they have enriched this body of knowledge from a variety of perspectives. The fruits of their endeavors are compiled in this *Handbook of Robotics*. More than 100 of

the world's leading experts have collaborated in producing this publication. Now, people who wish to commit themselves to robotics research can find a firm foundation to build upon. This Handbook is sure to be used to further advance robotics science, reinforce engineering education, and systematically compile knowledge that will innovate both society and industry.

The roles of humans and robots in an aging society pose an important issue for scientists and engineers to consider. Can robotics contribute to securing peace, prosperity, and a greater quality of life? This is still an open question. However, recent advances in personal robots, robotic home appliances, and humanoids suggest a paradigm shift from the industrial to the service sector. To realize this, robotics must be addressed from such viewpoints as the working infrastructure within society, psychophysiology, law, economy, insurance, ethics, art, design, drama, and sports science. Future robotics should be studied as a subject that envelops both humanity and technology. This Handbook offers a selected technical foundation upon which to advance such newly emerging fields of robotics. I look forward to continuing progress adding page after page of robot-based prosperity to future society.

Tokyo, September 2007

Hirochika Inoue

Foreword



Rodney Brooks

Panasonic Professor
of Robotics
Massachusetts Institute
of Technology

Robots have fascinated people for thousands of years. Those automata that were built before the 20th century did not connect sensing to action but rather operated through human agency or as repetitive machines. However, by the 1920s electronics had gotten to the stage that the first true robots that sensed the world and acted in it appropriately could be built. By 1950 we started to see descriptions of real robots appearing in popular magazines. By the 1960s industrial robots came onto the scene. Commercial pressures made them less and less responsive to their environments but faster and faster in what they did in their carefully engineered world. Then in the mid 1970s in France, Japan, and the USA we started to see robots rising again in a handful of research laboratories, and now we have arrived at a world-wide frenzy in research and the beginnings of large-scale deployment of intelligent robots throughout our world. This Handbook brings together the current state of robotics research in one place. It ranges from the mechanism of robots through sensing and perceptual processing, intelligence, action, and many application areas.

I have been more than fortunate to have lived with this revolution in robotics research over the last 30 years. As a teenager in Australia I built robots inspired by the tortoises of Walter described in the *Scientific American* in 1949 and 1950. When I arrived in Silicon Valley in 1977, just as the revolution in the personalization of computation was really coming into being, I instead turned to the much more obscure world of robots. In 1979 I was able to assist Hans Moravec at the Stanford Artificial Intelligence Lab ([SAIL](#)) as he coaxed his robot “The Cart” to navigate 20 m in 6 hours. Just 26 years later, in 2005, at the same laboratory, [SAIL](#), Sebastian Thrun and his team coaxed their robot to autonomously drive 200,000 m in 6 hours: four orders of magnitude improvement in a mere 26 years, which is slightly better than a doubling every 2 years. However, robots have not

just improved in speed, they have also increased in number. When I arrived at [SAIL](#) in 1977 we knew of three mobile robots operating in the world. Recently a company that I founded manufactured its 3,000,000th mobile robot, and the pace is increasing. Other aspects of robots have had similarly spectacular advances, although it is harder to provide such crisp numeric characterizations. In recent years we have gone from robots being too unaware of their surroundings that it was unsafe for people to share their workspace to robots that people can work with in close contact, and from robots that were totally unaware of people to robots that pick up on natural social cues from facial expressions to prosody in people’s voices. Recently robotics has crossed the divide between flesh and machines so that now we are seeing neuro-robotics ranging from prosthetic robotic extensions to rehabilitative robots for the disabled. And very recently robotics has become a respected contributor to research in cognitive science and neuroscience.

The research results chronicled in this volume give the key ideas that have enabled these spectacular advances. The editors, the part editors, and all the contributors have done a stellar job in bring this knowledge together in one place. Their efforts have produced a work that will provide a basis for much further research and development. Thank you, and congratulations to all who have labored on this pivotal book.

Some of the future robotics research will be incremental in nature, taking the state of the art and improving upon it. Other parts of future research will be more revolutionary, based on ideas that are antithetical to some of the ideas and current state of the art presented in this book.

As you study this volume and look for places to contribute to research through your own talents and hard work I want to alert you to capabilities or aspirations that I believe will make robots even more useful, more productive, and more accepted. I describe these capabilities in terms of the age at which a child has equivalent capabilities:

- the object-recognition capabilities of a 2-year-old child
- the language capabilities of a 4-year-old child

- the manual dexterity of a 6-year-old child
- the social understanding of an 8-year-old child

Each of these is a very difficult goal. However even small amounts of progress towards any one of these goals

will have immediate applications to robots out in the world. Good reading and best wishes as you contribute further to robotkind.

Cambridge, October 2007

Rodney Brooks

Preface

Reaching for the human frontier, robotics is vigorously engaged in the growing challenges of new emerging domains. Interacting, exploring, and working with humans, the new generation of robots will increasingly touch people and their lives. The credible prospect of practical robots among humans is the result of the scientific endeavor of a half a century of robotic developments that established robotics as a modern scientific discipline.

The undertaking of the *Springer Handbook of Robotics* was motivated by the rapid growth of the field. With the ever increasing amount of publications in journals, conference proceedings and monographs, it is difficult for those involved in robotics, particularly those who are just entering the field, to stay abreast of its wide range of developments. This task is made even more arduous by the very multidisciplinary nature of robotics.

The handbook follows preceding efforts in the 1980s and 1990s, which have brought valuable references to the robotics community: *Robot Motion: Planning and Control* (Brady, Hollerbach, Johnson, Lozano-Pérez, and Mason, MIT Press 1982), *Robotics Science* (Brady, MIT Press 1989), *The Robotics Review 1 and 2* (Khatib, Craig, and Lozano-Pérez, MIT Press 1989 and 1992). With the greater expansion of the robotics field and its increased outreach towards other scientific disciplines, the need for a comprehensive reference source combining basic and advanced developments has naturally become yet more urgent.

The volume is the result of the effort by a number of contributors who themselves are actively involved in robotics research in countries around the world. It has been a gigantic task to insightfully provide coverage of all the areas of robotics by such a motivated and versatile group of individuals committed to this endeavour.

The project started in May 2002 during a meeting the two of us had with Springer Director Engineering Europe Dieter Merkle and STAR Senior Editor Thomas Ditzinger. A year earlier, together with Frans Groen, we had launched the Springer Tracts in Advanced Robotics (STAR) series, which was rapidly establishing itself as an important medium for the timely dissemination of robotics research.

It was in this context that we took on this challenging task and enthusiastically began the planning to develop the technical structure and build the group of contributors. To capture the multiple dimensions of the field in its well-established academic core, ongoing research developments, and emerging applications, we conceived the handbook in a three-layer structure for a total of seven parts.

The first layer and part is devoted to the robotics foundations. The consolidated methodologies and technologies are grouped in the four parts of the second layer, covering robot structures, sensing and perception, manipulation and interfaces, mobile and distributed robotics. The third layer includes the advanced applications in the two parts on field and service robotics, and human-centered and life-like robotics, respectively.

To develop each of these parts, we envisioned the establishment of an editorial team which could coordinate the authors' contributions to the various chapters. A year later our seven-member editorial team was formed: David Orin, Frank Park, Henrik Christensen, Makoto Kaneko, Raja Chatila, Alex Zelinsky and Daniela Rus. With the commitment of such a group of distinguished scholars, the handbook was granted quality, span, and balance in the scientific areas.

By early 2005, we assembled an authorship of more than one-hundred-and-fifty contributors. An internal web site was created to facilitate inter-part and chapter cross-references, and to pace the schedule for the development of the project. The contents were carefully tuned over the following year, and especially during the two full-day workshops held in the spring of 2005 and 2006, well attended by most of the authors.

Each chapter was peer reviewed by at least three independent reviewers, typically involving the part editor, and two authors of related chapters; and in some cases, included external experts in the field. Two review cycles were necessary, and even three in some cases. During the process, a few more authors were recruited whenever it was deemed necessary. Most chapters were finalized by the end of the summer of 2007, and the project was completed by the early spring of 2008 – generating, by that time, a record of over 10 000 emails in our folders. The result is an impressive collection of 64 chapters

over the 7 parts, contributed by 165 authors, with more than 1650 pages, 950 illustrations and 5500 references.

We are deeply thankful to the authors for their intellectual contributions, as well as to the reviewers and part editors for their conscientious work. We are indebted to Werner Skolaut, the Senior Manager of Springer Handbooks in Science and Engineering, who soon became a devoted member of our team with his painstaking support to technically editing the authors' typescripts and linking the editors' work with the copy editing and production of the handbook. We also wish to acknowledge the highly professional work by the Le-T_EX staff, which re-typeset all the text, redrew and improved the many

illustrations, while timely interacting with the authors during the proof-reading of the material.

Six years after its conception the handbook comes to light. Beyond its tutorial value for our community, it is our hope that it will serve as a useful source to attract new researchers to robotics and inspire decades of vibrant progress in this fascinating field. The completion of every endeavor also brings new exciting challenges; at such times, our fellows are always reminded to . . . keep the gradient ;-)

Napoli and Stanford
April 2008

Bruno Siciliano
Oussama Khatib

About the Editors

Bruno Siciliano received his Doctorate degree in Electronic Engineering from the University of Naples in 1987. He is Professor of Control and Robotics, and Director of the **PRISMA** Lab in the Department of Computer and Systems Engineering at University of Naples. His current research is in force control, visual servoing, dual-arm/hand manipulation, lightweight flexible arms, human-robot interaction and service robotics. Professor Siciliano has co-authored 6 books and 5 edited volumes, 65 journal papers, 165 conference papers and book chapters, and has delivered 85 invited lectures and seminars at institutions around the world. He is Co-Editor of the Springer Tracts in Advanced Robotics (STAR) series, the Springer Handbook of Robotics, and has served on the Editorial Boards of prestigious journals, as well as Chair or Co-Chair for numerous international conferences. He is a Fellow of both **IEEE** and **ASME**. Professor Siciliano is the President of the **IEEE** Robotics and Automation Society (**RAS**), after serving as Vice President for Technical Activities and Vice President for Publications, as a Distinguished Lecturer, as a member of the Administrative Committee and of several other society committees.



Oussama Khatib received his Doctorate degree in Electrical Engineering from Sup'Aero, Toulouse, France, in 1980. He is Professor of Computer Science at Stanford University. His current research, which focuses on human-centered robotics, is concerned with human motion synthesis, humanoid robotics, haptic teleoperation, medical robotics, and human-friendly robot design. His research in these areas builds on a large body of studies he pursued over the past 25 years and published in over 200 contributions. Professor Khatib has delivered over 50 keynote presentations and several hundreds of colloquia and seminars at institutions around the world. He is Co-Editor of the STAR series, the Springer Handbook of Robotics, and has served on the Advisory and Editorial Boards of prestigious institutions and journals, as well as Chair or Co-Chair for numerous international conferences. He is a Fellow of **IEEE** and has served **RAS** as a Distinguished Lecturer and as a member of the Administrative Committee. Professor Khatib is the President of the International Foundation of Robotics Research (**IFRR**) and a recipient of the Japan Robot Association (JARA) Award in Research and Development.



About the Part Editors



David E. Orin

The Ohio State University
Department of Electrical Engineering
Columbus, OH, USA
orin.1@osu.edu

Part A

David E. Orin received his PhD degree in Electrical Engineering from The Ohio State University in 1976. From 1976 to 1980 he taught at Case Western Reserve University. Since 1981, he has been at The Ohio State University, where he is currently a Professor of Electrical and Computer Engineering. His current work focuses on dynamic movement in biped locomotion. He has made many contributions to robot dynamics and legged locomotion, and he has over 125 publications. He has received a number of educational awards from his institution. He is an IEEE Fellow and has served on the program committee for several international conferences. He has received the Distinguished Service Award in recognition of his service for the IEEE Robotics and Automation Society, such as Vice President for Finance, Secretary, member of the Administrative Committee, and co-chair of the Fellow Evaluation Committee.

Frank C. Park

Seoul National University
Mechanical and Aerospace Engineering
Seoul, Korea
fcp@snu.ac.kr



Part B

Frank C. Park received the PhD degree in Applied Mathematics from Harvard University in 1991. From 1991 to 1995 he was an Assistant Professor of Mechanical and Aerospace Engineering at the University of California, Irvine. Since 1995 he has been at the School of Mechanical and Aerospace Engineering at Seoul National University, where he is currently full professor. His primary research interests in robotics include robot mechanics, planning, and control, robot design and structures, and industrial robotics. His other research interests include nonlinear systems theory, differential geometry and its applications, and related areas of applied mathematics. He has served the IEEE Robotics and Automation Society as a Secretary and as a Senior Editor of the IEEE Transactions on Robotics.

Henrik I. Christensen

Georgia Institute of Technology
Robotics and Intelligent Machines @ GT
Atlanta, GA, USA
hic@cc.gatech.edu



Part C

Henrik I. Christensen is the KUKA Chair of Robotics and Director of Robotics at Georgia Institute of Technology, Atlanta, GA. He received the MS and PhD degrees from Aalborg University in 1987 and 1990, respectively. He has held positions in Denmark, Sweden, and USA. He has published more than 250 contributions across vision, robotics, and AI. Results from his research have been commercialized through a number of major companies and 4 spin-offs. He served as the founding coordinator of the European Robotics Research Network (EURON). He has participated as a senior organizer in more than 50 different conferences and workshops. He is a Fellow of the International Foundation of Robotics Research, an Editorial Board member of the STAR series, and has served on the editorial board of several leading journals in the field. He has been a Distinguished Lecturer of the IEEE Robotics and Automation Society.



Makoto Kaneko

Osaka University
Department of Mechanical Engineering
Graduate School of Engineering
Suita, Japan
mk@mech.eng.osaka-u.ac.jp

Part D

Makoto Kaneko received the MS and PhD degrees in Mechanical Engineering from Tokyo University in 1978 and 1981, respectively. From 1981 to 1990 he was Researcher at the Mechanical Engineering Laboratory, from 1990 to 1993 an Associate Professor at Kyushu Institute of Technology, from 1993 to 2006 Professor at Hiroshima University, and in 2006 became a Professor at Osaka University. His research interests include tactile-based active sensing, grasping strategy, hyper human technology and its application to medical diagnosis, and his work has received 17 awards. He is an Editorial Board member of the STAR series and has served as chair or co-chair for several international conferences. He is an IEEE Fellow. He has served the IEEE Robotics and Automation Society as a Vice-President for Member Activities and as a Technical Editor of the IEEE Transactions on Robotics and Automation.



Raja Chatila

LAAS-CNRS
Toulouse, France
raja.chatila@laas.fr

Part E

Raja Chatila received his PhD degree from the University of Toulouse in 1981. He is currently Director of LAAS-CNRS (Laboratory of Systems Analysis and Architecture), Toulouse, France where he is since 1983. He was an invited professor at Tsukuba University in 1997. His research work encompasses several aspects in field, planetary, aerial and service robotics, cognitive robotics, learning, human-robot interaction, and networked robotics. He is author of over 150 international publications on these topics. He is a Fellow of the International Foundation of Robotics Research. He has served on the editorial boards of several leading publications, including the STAR series, and as chair or co-chair for several international conferences. He has served the IEEE Robotics and Automation Society as a member of the Administrative Committee, as an Associate Editor of the IEEE Transactions on Robotics and Automation and as a Distinguished Lecturer. He is a member of IEEE, ACM and AAAI, and a member of various national and international boards and evaluation committees.

Alexander Zelinsky

Commonwealth Scientific and Industrial
Research Organisation (CSIRO)
ICT Centre
Epping, NSW, Australia
alex.zelinsky@csiro.au



Part F

Alexander Zelinsky is the Director of the Information and Communication Technologies Centre at the CSIRO. Before joining CSIRO, he was CEO and Founder of Seeing Machines Pty Limited and a Professor at the Australian National University, in the Research School of Information Sciences and Engineering. He is a well-known scientist specializing in robotics and computer vision, he is widely recognized as an innovator in human-machine interaction, and he has over 100 publications in the field. His work has received both national and international awards. He has served on the editorial boards of two leading publications, and on the program committees of several international conferences. He is an IEEE Fellow and has served the IEEE Robotics and Automation Society as a member of the Administrative Committee and as Vice-President for Industrial Activities.

Daniela Rus

Massachusetts Institute of Technology
CSAIL Center for Robotics
Cambridge, MA, USA
rus@csail.mit.edu



Part G

Daniela Rus received her PhD degree in Computer Science from Cornell University in 1992. From 1994 to 2003 she taught at Dartmouth, Hanover. Since 2004 she has been at MIT, where she is currently a Professor of Electrical Engineering and Computer Science. She co-directs the CSAIL Center for robotics and is an Associate Director of CSAIL. Her research interests center on distributed robotics and mobile computing, and she has published extensively in the field. Her work in robotics aims to develop self-organizing systems and spans the spectrum from novel mechanical design and experimental platforms to developing and analyzing algorithms for locomotion, manipulation, and group control. She has received a number of awards, including the MacArthur Fellow. She has been on the program committees of several international conferences, and has served the IEEE Robotics and Automation Society as Education Co-Chair.

List of Contributors

Jorge Angeles

McGill University
Department of Mechanical Engineering
and Centre for Intelligent Machines
817 Sherbrooke St. W.
Montreal, Quebec H3A 2K6, Canada
e-mail: angeles@cim.mcgill.ca

Gianluca Antonelli

Università degli Studi di Cassino
Dipartimento di Automazione, Ingegneria
dell'Informazione e Matematica Industriale
Via G. Di Biasio 43
03043 Cassino, Italy
e-mail: antonelli@unicas.it

Fumihito Arai

Tohoku University
Department of Bioengineering and Robotics
6-6-01 Aoba-yama
980-8579 Sendai, Japan
e-mail: arai@imech.mech.tohoku.ac.jp

Michael A. Arbib

University of Southern California
Computer, Neuroscience and USC Brain Project
Los Angeles, CA 90089-2520, USA
e-mail: arbib@usc.edu

Antonio Bicchi

Università degli Studi di Pisa
Centro Interdipartimentale di Ricerca
"Enrico Piaggio" e Dipartimento
di Sistemi Elettrici e Automazione
Via Diotisalvi 2
56125 Pisa, Italy
e-mail: bicchi@ing.unipi.it

Aude Billard

Ecole Polytechnique Federale de Lausanne (EPFL)
Learning Algorithms and Systems Laboratory (LASA)
STI-I2S-LASA
1015 Lausanne, Switzerland
e-mail: aude.billard@epfl.ch

John Billingsley

University of Southern Queensland
Faculty of Engineering and Surveying
Toowoomba QLD 4350, Australia
e-mail: billings@usq.edu.au

Wayne Book

Georgia Institute of Technology
G.W. Woodruff School of Mechanical Engineering
771 Ferst Drive
Atlanta, GA 30332-0405, USA
e-mail: wayne.book@me.gatech.edu

Cynthia Breazeal

Massachusetts Institute of Technology
The Media Lab
20 Ames St.
Cambridge, MA 02139, USA
e-mail: cynthiab@media.mit.edu

Oliver Brock

University of Massachusetts
Robotics and Biology Laboratory
140 Governors Drive
Amherst, MA 01003, USA
e-mail: oli@cs.umass.edu

Alberto Broggi

Università degli Studi di Parma
Dipartimento di Ingegneria dell'Informazione
Viale delle Scienze 181A
43100 Parma, Italy
e-mail: broggi@ce.unipr.it

Heinrich H. Bülthoff

Max-Planck-Institut für biologische Kybernetik
Kognitive Humanpsychophysik
Spemannstr. 38
72076 Tübingen, Germany
e-mail: heinrich.buelthoff@tuebingen.mpg.de

Joel W. Burdick

California Institute of Technology
Mechanical Engineering Department
1200 E. California Blvd.
Pasadena, CA 91125, USA
e-mail: jwb@robotics.caltech.edu

Wolfram Burgard

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Georges-Koehler-Allee 079
79110 Freiburg, Germany
e-mail: burgard@informatik.uni-freiburg.de

Zack Butler

Rochester Institute of Technology
Department of Computer Science
102 Lomb Memorial Dr.
Rochester, NY 14623, USA
e-mail: zjb@cs.rit.edu

Fabrizio Caccavale

Università degli Studi della Basilicata
Dipartimento di Ingegneria e Fisica dell'Ambiente
Via dell'Ateneo Lucano 10
85100 Potenza, Italy
e-mail: fabrizio.caccavale@unibas.it

Sylvain Calinon

Ecole Polytechnique Federale de Lausanne (EPFL)
Learning Algorithms and Systems Laboratory (LASA)
STI-I2S-LASA
1015 Lausanne, Switzerland
e-mail: sylvain.calinon@epfl.ch

Guy Campion

Université Catholique de Louvain
Centre d'Ingénierie des Systèmes d'Automatique
et de Mécanique Appliquée
4 Avenue G. Lemaître
1348 Louvain-la-Neuve, Belgium
e-mail: guy.campion@uclouvain.be

Raja Chatila

LAAS-CNRS
7 Avenue du Colonel Roche
31077 Toulouse, France
e-mail: raja.chatila@laas.fr

François Chaumette

INRIA/IRISA
Campus de Beaulieu
35042 Rennes, France
e-mail: francois.chaumette@irisa.fr

Stefano Chiaverini

Università degli Studi di Cassino
Dipartimento di Automazione, Ingegneria
dell'Informazione e Matematica Industriale
Via G. Di Biasio 43
03043 Cassino, Italy
e-mail: chiaverini@unicas.it

Nak Young Chong

Japan Advanced Institute of Science
and Technology (JAIST)
School of Information Science
1-1 Asahidai, Nomi
923-1292 Ishikawa, Japan
e-mail: nakyoung@jaist.ac.jp

Howie Choset

Carnegie Mellon University
The Robotics Institute
5000 Forbes Ave.
Pittsburgh, PA 15213, USA
e-mail: choset@cs.cmu.edu

Henrik I. Christensen

Georgia Institute of Technology
Robotics and Intelligent Machines @ GT
Atlanta, GA 30332-0760, USA
e-mail: hic@cc.gatech.edu

Wankyun Chung

POSTECH
Department of Mechanical Engineering
San 31 Hyojading
Pohang 790-784, Korea
e-mail: wkchung@postech.ac.kr

Woojin Chung

Korea University
Department of Mechanical Engineering
Anam-dong, Sungbuk-ku
Seoul 136-701, Korea
e-mail: smartrobot@korea.ac.kr

J. Edward Colgate

Northwestern University
 Department of Mechanical Engineering
 Segal Design Institute
 2145 Sheridan Rd.
 Evanston, IL 60208, USA
 e-mail: colgate@northwestern.edu

Peter Corke

Commonwealth Scientific
 and Industrial Research Organisation (CSIRO)
 ICT Centre
 PO Box 883
 Kenmore QLD 4069, Australia
 e-mail: peter.corke@csiro.au

Jock Cunningham

Commonwealth Scientific
 and Industrial Research Organisation (CSIRO)
 Division of Exploration and Mining
 PO Box 883
 Kenmore QLD 4069, Australia
 e-mail: jock.cunningham@csiro.au

Mark R. Cutkosky

Stanford University
 Mechanical Engineering
 Building 560, 424 Panama Mall
 Stanford, CA 94305-2232, USA
 e-mail: cutkosky@stanford.edu

Kostas Daniilidis

University of Pennsylvania
 Department of Computer and Information Science
 GRASP Laboratory
 3330 Walnut Street
 Philadelphia, PA 19104, USA
 e-mail: kostas@cis.upenn.edu

Paolo Dario

Scuola Superiore Sant'Anna
 ARTS Lab e CRIM Lab
 Piazza Martiri della Libertà 33
 56127 Pisa, Italy
 e-mail: paolo.dario@sssup.it

Alessandro De Luca

Università degli Studi di Roma "La Sapienza"
 Dipartimento di Informatica
 e Sistemistica "A. Ruberti"
 Via Ariosto 25
 00185 Roma, Italy
 e-mail: deluca@dis.uniroma1.it

Joris De Schutter

Katholieke Universiteit Leuven
 Department of Mechanical Engineering
 Celestijnenlaan 300, Box 02420
 3001 Leuven-Heverlee, Belgium
 e-mail: joris.deschutter@mech.kuleuven.be

Rüdiger Dillmann

Universität Karlsruhe
 Institut für Technische Informatik
 Haid-und-Neu-Str. 7
 76131 Karlsruhe, Germany
 e-mail: dillmann@ira.uka.de

Lixin Dong

ETH Zentrum
 Institute of Robotics and Intelligent Systems
 Tannenstr. 3
 8092 Zürich, Switzerland
 e-mail: ldong@ethz.ch

Gregory Dudek

McGill University
 Department of Computer Science
 3480 University Street
 Montreal, QC H3Y 3H4, Canada
 e-mail: dudek@cim.mcgill.ca

Mark Dunn

University of Southern Queensland
 National Centre for Engineering in Agriculture
 Toowoomba QLD 4350, Australia
 e-mail: mark.dunn@usq.edu.au

Hugh Durrant-Whyte

University of Sydney
 ARC Centre of Excellence for Autonomous Systems
 Australian Centre for Field Robotics (ACFR)
 Sydney NSW 2006, Australia
 e-mail: hugh@acfr.usyd.edu.au

Jan-Olof Eklundh

KTH Royal Institute of Technology
Teknikringen 14
10044 Stockholm, Sweden
e-mail: joe@nada.kth.se

Aydan M. Erkmen

Middle East Technical University
Department of Electrical Engineering
Ankara, 06531, Turkey
e-mail: aydan@metu.edu.tr

Bernard Espiau

INRIA Rhône-Alpes
38334 Saint-Ismier, France
e-mail: bernard.espiau@inria.fr

Roy Featherstone

The Australian National University
Department of Information Engineering
RSISE Building 115
Canberra ACT 0200, Australia
e-mail: roy.featherstone@anu.edu.au

Eric Feron

Georgia Institute of Technology
School of Aerospace Engineering
270 Ferst Drive
Atlanta, GA 30332-0150, USA
e-mail: feron@gatech.edu

Gabor Fichtinger

Queen's University
School of Computing
#725 Goodwin Hall, 25 Union St.
Kingston, ON K7L 3N6, Canada
e-mail: gabor@cs.queensu.ca

Paolo Fiorini

Università degli Studi di Verona
Dipartimento di Informatica
Strada le Grazie 15
37134 Verona, Italy
e-mail: paolo.fiorini@univr.it

Robert B. Fisher

University of Edinburgh
School of Informatics
James Clerk Maxwell Building, Mayfield Road
Edinburgh, EH9 3JZ, UK
e-mail: rbr@inf.ed.ac.uk

Paul Fitzpatrick

Italian Institute of Technology
Robotics, Brain, and Cognitive
Sciences Department
Via Morego 30
16163 Genova, Italy
e-mail: paul.fitzpatrick@iit.it

Dario Floreano

Ecole Polytechnique Federale de Lausanne (EPFL)
Laboratory of Intelligent Systems
EPFL-STI-I2S-LIS
1015 Lausanne, Switzerland
e-mail: dario.floreano@epfl.ch

Thor I. Fossen

Norwegian University of Science
and Technology (NTNU)
Department of Engineering Cybernetics
Trondheim, 7491, Norway
e-mail: fossen@ieee.org

Li-Chen Fu

National Taiwan University
Department of Electrical Engineering
Taipei, 106, Taiwan, R.O.C.
e-mail: lichen@ntu.edu.tw

Maxime Gautier

Université de Nantes
IRCCyN, ECN
1 Rue de la Noë
44321 Nantes, France
e-mail: maxime.gautier@ircryn.ec-nantes.fr

Martin A. Giese

University of Wales
Department of Psychology
Penrallt Rd.
Bangor, LL 57 2AS, UK
e-mail: martin.giese@uni-tuebingen.de

Ken Goldberg

University of California at Berkeley
 Department of Industrial Engineering
 and Operations Research
 4141 Etcheverry Hall
 Berkeley, CA 94720-1777, USA
 e-mail: goldberg@jeor.berkeley.edu

Clément Gosselin

Université Laval
 Département de Genie Mecanique
 Quebec, QC G1K 7P4, Canada
 e-mail: gosselin@gmc.ulaval.ca

Agnès Guillot

Université Pierre et Marie Curie – CNRS
 Institut des Systèmes Intelligents et de Robotique
 4 Place Jussieu
 75252 Paris, France
 e-mail: agnes.guillot@lip6.fr

Martin Hägele

Fraunhofer IPA
 Robot Systems
 Nobelstr. 12
 70569 Stuttgart, Germany
 e-mail: mmh@ipa.fhg.de

Gregory D. Hager

Johns Hopkins University
 Department of Computer Science
 3400 N. Charles St.
 Baltimore, MD 21218, USA
 e-mail: hager@cs.jhu.edu

David Hainsworth

Commonwealth Scientific
 and Industrial Research Organisation (CSIRO)
 Division of Exploration and Mining
 PO Box 883
 Kenmore QLD 4069, Australia
 e-mail: david.hainsworth@csiro.au

William R. Hamel

University of Tennessee
 Mechanical, Aerospace,
 and Biomedical Engineering
 414 Dougherty Engineering Building
 Knoxville, TN 37996-2210, USA
 e-mail: whamel@utk.edu

Blake Hannaford

University of Washington
 Department of Electrical Engineering
 Box 352500
 Seattle, WA 98195-2500, USA
 e-mail: blake@ee.washington.edu

Kensuke Harada

National Institute of Advanced Industrial Science
 and Technology (AIST)
 Intelligent Systems Research Institute
 1-1-1 Umezono
 305-8568 Tsukuba, Japan
 e-mail: kensuke.harada@aist.go.jp

Martial Hebert

Carnegie Mellon University
 The Robotics Institute
 5000 Forbes Ave.
 Pittsburgh, PA 15213, USA
 e-mail: hebert@ri.cmu.edu

Thomas C. Henderson

University of Utah
 School of Computing
 50 S. Central Campus Dr. 3190 MEB
 Salt Lake City, UT 84112, USA
 e-mail: tch@cs.utah.edu

Joachim Hertzberg

Universität Osnabrück
 Institut für Informatik
 Albrechtstr. 28
 54076 Osnabrück, Germany
 e-mail: hertzberg@informatik.uni-osnabrueck.de

Hirohisa Hirukawa

National Institute of Advanced Industrial Science
and Technology (AIST)
Intelligent Systems Research Institute
1-1-1 Umezono
305-8568 Tsukuba, Japan
e-mail: hiro.hirukawa@aist.go.jp

Gerd Hirzinger

Deutsches Zentrum für Luft- und Raumfahrt (DLR)
Oberpfaffenhofen
Institut für Robotik und Mechatronik
Münchner Str. 20
82230 Wessling, Germany
e-mail: gerd.hirzinger@dlr.de

John Hollerbach

University of Utah
School of Computing
50 S. Central Campus Dr.
Salt Lake City, UT 84112, USA
e-mail: jmh@cs.utah.edu

Robert D. Howe

Harvard University
Division of Engineering and Applied Sciences
Pierce Hall, 29 Oxford St.
Cambridge, MA 02138, USA
e-mail: howe@seas.harvard.edu

Su-Hau Hsu[†]

National Taiwan University
Taipei, Taiwan

Phil Husbands

University of Sussex
Department of Informatics
Falmer, Brighton BN1 9QH, UK
e-mail: philh@sussex.ac.uk

Seth Hutchinson

University of Illinois
Department of Electrical and Computer
Engineering
Urbana, IL 61801, USA
e-mail: seth@uiuc.edu

Adam Jacoff

National Institute of Standards and Technology
Intelligent Systems Division
100 Bureau Drive
Gaithersburg, MD 20899, USA
e-mail: adam.jacoff@nist.gov

Michael Jenkin

York University
Computer Science and Engineering
4700 Keel St.
Toronto, Ontario M3J 1P3, Canada
e-mail: jenkin@cse.yorku.ca

Eric N. Johnson

Georgia Institute of Technology
Daniel Guggenheim School of
Aerospace Engineering
270 Ferst Drive
Atlanta, GA 30332-0150, USA
e-mail: eric.johnson@ae.gatech.edu

Shuuji Kajita

National Institute of Advanced Industrial Science
and Technology (AIST)
Intelligent Systems Research Institute
1-1-1 Umezono
305-8568 Tsukuba, Japan
e-mail: s.kajita@aist.go.jp

Makoto Kaneko

Osaka University
Department of Mechanical Engineering
Graduate School of Engineering
2-1 Yamadaoka
565-0871 Suita, Osaka, Japan
e-mail: mk@mech.eng.osaka-u.ac.jp

Sung-Chul Kang

Korea Institute of Science and Technology
Cognitive Robotics Research Center
Hawolgok-dong 39-1, Sungbuk-ku
Seoul 136-791, Korea
e-mail: kasch@kist.re.kr

Imin Kao

State University of New York at Stony Brook
 Department of Mechanical Engineering
 Stony Brook, NY 11794-2300, USA
 e-mail: imin.kao@stonybrook.edu

Lydia E. Kavradi

Rice University
 Department of Computer Science, MS 132
 6100 Main Street
 Houston, TX 77005, USA
 e-mail: kavraki@rice.edu

Homayoon Kazerooni

University of California at Berkeley
 Berkeley Robotics and Human Engineering
 Laboratory
 5124 Etcheverry Hall
 Berkeley, CA 94720-1740, USA
 e-mail: kazerooni@berkeley.edu

Charles C. Kemp

Georgia Institute of Technology
 and Emory University
 The Wallace H. Coulter Department
 of Biomedical Engineering
 313 Ferst Drive
 Atlanta, GA 30332-0535, USA
 e-mail: charlie.kemp@bme.gatech.edu

Wisama Khalil

Université de Nantes
 IRCCyN, ECN
 1 Rue de la Noë
 44321 Nantes, France
 e-mail: wisama.khalil@ircryn.ec-nantes.fr

Oussama Khatib

Stanford University
 Department of Computer Science
 Artificial Intelligence Laboratory
 Stanford, CA 94305-9010, USA
 e-mail: khatib@cs.stanford.edu

Lindsay Kleeman

Monash University
 Department of Electrical and Computer Systems
 Engineering
 Department of ECSEng
 Monash VIC 3800, Australia
 e-mail: kleeman@eng.monash.edu.au

Tetsunori Kobayashi

Waseda University
 Department of Computer Science
 3-4-1 Okubo, Shinjuku-ku
 169-8555 Tokyo, Japan
 e-mail: koba@waseda.jp

Kurt Konolige

SRI International
 Artificial Intelligence Center
 333 Ravenswood Ave.
 Menlo Park, CA 94025, USA
 e-mail: konolige@ai.sri.com

David Kortenkamp

TRAC Labs Inc.
 1012 Hercules Drive
 Houston, TX 77058, USA
 e-mail: korten@traclabs.com

Kazuhiro Kosuge

Tohoku University
 Department of Bioengineering and Robotics
 Graduate School of Engineering
 6-6-01 Aoba-yama
 980-8579 Sendai, Japan
 e-mail: kosuge@irs.mech.tohoku.ac.jp

Roman Kuc

Yale University
 Department of Electrical Engineering
 10 Hillhouse Ave
 New Haven, CT 06520-8267, USA
 e-mail: kuc@yale.edu

James Kuffner

Carnegie Mellon University
 The Robotics Institute
 5000 Forbes Ave.
 Pittsburgh, PA 15213, USA
 e-mail: kuffner@cs.cmu.edu

Vijay Kumar

University of Pennsylvania
Department of Mechanical Engineering
and Applied Mechanics
220 S. 33rd Street
Philadelphia, PA 19104-6315, USA
e-mail: kumar@grasp.upenn.edu

Florent Lamiroux

LAAS-CNRS
7 Avenue du Colonel Roche
31077 Toulouse, France
e-mail: florent@laas.fr

Jean-Paul Laumond

LAAS-CNRS
7 Avenue du Colonel Roche
31077 Toulouse, France
e-mail: jpl@laas.fr

Steven M. LaValle

University of Illinois
Department of Computer Science
201 N. Goodwin Ave, 3318 Siebel Center
Urbana, IL 61801, USA
e-mail: lavalle@cs.uiuc.edu

John J. Leonard

Massachusetts Institute of Technology
Department of Mechanical Engineering
5-214 77 Massachusetts Ave
Cambridge, MA 02139, USA
e-mail: jleonard@mit.edu

Kevin Lynch

Northwestern University
Mechanical Engineering Department
2145 Sheridan Road
Evanston, IL 60208, USA
e-mail: kmlynch@northwestern.edu

Alan M. Lytle

National Institute of Standards and Technology
Construction Metrology and Automation Group
100 Bureau Drive
Gaithersburg, MD 20899, USA
e-mail: alan.lytle@nist.gov

Maja J. Mataric

University of Southern California
Computer Science Department
3650 McClintock Avenue
Los Angeles, CA 90089, USA
e-mail: mataric@usc.edu

Yoshio Matsumoto

Osaka University
Department of Adaptive Machine Systems
Graduate School of Engineering
565-0871 Suita, Osaka, Japan
e-mail: matsumoto@ams.eng.osaka-u.ac.jp

J. Michael McCarthy

University of California at Irvine
Department of Mechanical and Aerospace
Engineering
Irvine, CA 92697, USA
e-mail: jmmccart@uci.edu

Claudio Melchiorri

Università degli Studi di Bologna
Dipartimento di Elettronica Informatica
e Sistemistica
Via Risorgimento 2
40136 Bologna, Italy
e-mail: claudio.melchiorri@unibo.it

Arianna Menciassi

Scuola Superiore Sant'Anna
CRIM Lab
Piazza Martiri della Libertà 33
56127 Pisa, Italy
e-mail: arianna@sssup.it

Jean-Pierre Merlet

INRIA Sophia-Antipolis
2004 Route des Lucioles
06902 Sophia-Antipolis, France
e-mail: jean-pierre.merlet@sophia.inria.fr

Giorgio Metta

Italian Institute of Technology
Department of Robotics, Brain and Cognitive
Sciences
Via Morego 30
16163 Genova, Italy
e-mail: pasa@liralab.it

Jean-Arcady Meyer

Université Pierre et Marie Curie – CNRS
 Institut des Systèmes Intelligents et de Robotique
 4 Place Jussieu
 75252 Paris, France
 e-mail: jean-arcady.meyer@lip6.fr

François Michaud

Université de Sherbrooke
 Department of Electrical Engineering
 and Computer Engineering
 2500 Boulevard Université
 Sherbrooke, Québec J1K 2R1, Canada
 e-mail: francois.michaud@usherbrooke.ca

David P. Miller

University of Oklahoma
 School of Aerospace and Mechanical Engineering
 865 Asp Ave.
 Norman, OK 73019, USA
 e-mail: dpmiller@ou.edu

Javier Minguez

Universidad de Zaragoza
 Departamento de Informática e Ingeniería de
 Sistemas
 Centro Politécnico Superior
 Edificio Ada Byron, Maria de Luna 1
 Zaragoza 50018, Spain
 e-mail: jminguez@unizar.es

Pascal Morin

INRIA Sophia-Antipolis
 2004 Route des Lucioles
 06902 Sophia-Antipolis, France
 e-mail: pascal.morin@inria.fr

Robin R. Murphy

University of South Florida
 Computer Science and Engineering
 4202 E. Fowler Ave ENB342
 Tampa, FL 33620-5399, USA
 e-mail: murphy@cse.usf.edu

Daniele Nardi

Università degli Studi di Roma "La Sapienza"
 Dipartimento di Informatica e Sistemistica
 "A. Ruberti"
 Via Ariosto 25
 00185 Roma, Italy
 e-mail: nardi@dis.uniroma1.it

Bradley J. Nelson

ETH Zentrum
 Institute of Robotics and Intelligent Systems
 Tannenstr. 3
 8092 Zürich, Switzerland
 e-mail: bnelson@ethz.ch

Günter Niemeyer

Stanford University
 Department of Mechanical Engineering
 Design Group, Terman Engineering Center
 Stanford, CA 94305-4021, USA
 e-mail: gunter.niemeyer@stanford.edu

Klas Nilsson

Lund University
 Department of Computer Science
 Ole Römers väg 3
 22100 Lund, Sweden
 e-mail: klas@cs.lu.se

Stefano Nolfi

Consiglio Nazionale delle Ricerche (CNR)
 Istituto di Scienze e Tecnologie della Cognizione
 Via S. Martino della Battaglia 44
 00185 Roma, Italy
 e-mail: stefano.nolfi@istc.cnr.it

Illah R. Nourbakhsh

Carnegie Mellon University
 The Robotics Institute
 5000 Forbes Ave.
 Pittsburgh, PA 15213, USA
 e-mail: illah@cs.cmu.edu

Jonathan B. O'Brien

University of New South Wales
 School of Civil and Environmental Engineering
 Sydney 2052, Australia
 e-mail: j.obrien@unsw.edu.au

Allison M. Okamura

The Johns Hopkins University
Department of Mechanical Engineering
3400 N. Charles Street
Baltimore, MD 21218, USA
e-mail: aokamura@jhu.edu

Fiorella Operto

Scuola di Robotica
Piazza Monastero 4
16149 Sampierdarena, Genova, Italy
e-mail: operto@scuoladirobotica.it

David E. Orin

The Ohio State University
Department of Electrical Engineering
2015 Neil Avenue
Columbus, OH 43210, USA
e-mail: orin.1@osu.edu

Giuseppe Oriolo

Università degli Studi di Roma "La Sapienza"
Dipartimento di Informatica e Sistemistica
"A. Ruberti"
Via Ariosto 25
00185 Roma, Italy
e-mail: oriolo@dis.uniroma1.it

Michel Parent

INRIA Rocquencourt
78153 Le Chesnay, France
e-mail: michel.parent@inria.fr

Frank C. Park

Seoul National University
Mechanical and Aerospace Engineering
Seoul 51-742, Korea
e-mail: fcp@snu.ac.kr

Lynne E. Parker

University of Tennessee
Department of Electrical Engineering
and Computer Science
1122 Volunteer Blvd.
Knoxville, TN 37996-3450, USA
e-mail: parker@eecs.utk.edu

Michael A. Peshkin

Northwestern University
Department of Mechanical Engineering
2145 Sheridan Road
Evanston, IL 60208, USA
e-mail: peshkin@northwestern.edu

J. Norberto Pires

Universidade de Coimbra
Departamento de Engenharia Mecânica
Polo II
Coimbra 3030, Portugal
e-mail: norberto@robotics.dem.uc.pt

Erwin Prassler

Fachhochschule Bonn-Rhein-Sieg
Fachbereich Informatik
Grantham-Allee 20
53757 Sankt Augustin, Germany
e-mail: erwin.prassler@fh-brs.de

Domenico Prattichizzo

Università degli Studi di Siena
Dipartimento di Ingegneria dell'Informazione
Via Roma 56
53100 Siena, Italy
e-mail: prattichizzo@ing.unisi.it

Carsten Preusche

Deutsches Zentrum für Luft- und Raumfahrt (DLR)
Oberpfaffenhofen
Institut für Robotik und Mechatronik
Münchner Str. 20
82234 Wessling, Germany
e-mail: carsten.preusche@dlr.de

William R. Provancher

University of Utah
Department of Mechanical Engineering
50 S. Central Campus, 2120 MEB
Salt Lake City, UT 84112-9208, USA
e-mail: wil@mech.utah.edu

David J. Reinkensmeyer

University of California at Irvine
Mechanical and Aerospace Engineering
4200 Engineering Gateway
Irvine, CA 92617-3975, USA
e-mail: dreinken@uci.edu

Alfred Rizzi

Boston Dynamics
78 Fourth Ave
Waltham, MA 02451, USA
e-mail: arizzi@bostondynamics.com

Jonathan Roberts

Commonwealth Scientific
and Industrial Research Organisation (CSIRO)
ICT Centre, Autonomous Systems Laboratory
P.O. Box 883
Kenmore QLD 4069, Australia
e-mail: jonathan.roberts@csiro.au

Daniela Rus

Massachusetts Institute of Technology
CSAIL Center for Robotics
32 Vassar Street
Cambridge, MA 01239, USA
e-mail: rus@csail.mit.edu

Kamel S. Saidi

National Institute of Standards and Technology
Building and Fire Research Laboratory
100 Bureau Drive
Gaithersburg, MD 20899, USA
e-mail: kamel.saidi@nist.gov

Claude Samson

INRIA Sophia-Antipolis
2004 Route des Lucioles
06902 Sophia-Antipolis, France
e-mail: claudio.samson@inria.fr

Stefan Schaal

University of Southern California
Computer Science and Neuroscience
3710 S. McClintock Ave.
Los Angeles, CA 90089-2905, USA
e-mail: sschaal@usc.edu

Victor Scheinman

Stanford University
Department of Mechanical Engineering
Stanford, CA 94305, USA
e-mail: vds@stanford.edu

James Schmiedeler

The Ohio State University
Department of Mechanical Engineering
E307 Scott Laboratory, 201 West 19th Ave
Columbus, OH 43210, USA
e-mail: schmiedeler.2@osu.edu

Bruno Siciliano

Università degli Studi di Napoli Federico II
Dipartimento di Informatica e Sistemistica,
PRISMA Lab
Via Claudio 21
80125 Napoli, Italy
e-mail: siciliano@unina.it

Roland Siegwart

ETH Zentrum
Department of Mechanical and Process
Engineering
Tannenstr. 3, CLA E32
8092 Zürich, Switzerland
e-mail: rsiegwart@ethz.ch

Reid Simmons

Carnegie Mellon University
The Robotics Institute
School of Computer Science
5000 Forbes Ave.
Pittsburgh, PA 15241, USA
e-mail: reids@cs.cmu.edu

Dezhen Song

Texas A&M University
Department of Computer Science
H.R. Bright Building
College Station, TX 77843, USA
e-mail: dzsong@cs.tamu.edu

Gaurav S. Sukhatme

University of Southern California
Department of Computer Science
3710 South McClintock Ave
Los Angeles, CA 90089-2905, USA
e-mail: gaurav@usc.edu

Satoshi Tadokoro

Tohoku University
Graduate School of Information Sciences
6-6-01 Aoba-yama
980-8579 Sendai, Japan
e-mail: tadokoro@rm.is.tohoku.ac.jp

Atsuo Takanishi

Waseda University
Department of Modern Mechanical Engineering
3-4-1 Ookubo, Shinjuku-ku
169-8555 Tokyo, Japan
e-mail: takanisi@waseda.jp

Russell H. Taylor

The Johns Hopkins University
Department of Computer Science
Computational Science and Engineering Building
1-127, 3400 North Charles Street
Baltimore, MD 21218, USA
e-mail: rht@jhu.edu

Charles E. Thorpe

Carnegie Mellon University in Qatar
Qatar Office SMC 1070
5032 Forbes Ave.
Pittsburgh, PA 15289, USA
e-mail: thorpe@qatar.cmu.edu

Sebastian Thrun

Stanford University
Department of Computer Science
Artificial Intelligence Laboratory
Stanford, CA 94305-9010, USA
e-mail: thrun@stanford.edu

James P. Trevelyan

The University of Western Australia
School of Mechanical Engineering
35 Stirling Highway, Crawley
Perth Western Australia 6009, Australia
e-mail: james.trevelyan@uwa.edu.au

Jeffrey C. Trinkle

Rensselaer Polytechnic Institute
Department of Computer Science
Troy, NY 12180-3590, USA
e-mail: trink@cs.rpi.edu

Masaru Uchiyama

Tohoku University
Department of Aerospace Engineering
6-6-01 Aoba-yama
980-8579 Sendai, Japan
e-mail: uchiyama@space.mech.tohoku.ac.jp

H.F. Machiel Van der Loos

University of British Columbia
Department of Mechanical Engineering
6250 Applied Science Lane
Vancouver, BC V6T 1Z4, Canada
e-mail: vdl@mech.ubc.ca

Patrick van der Smagt

Deutsches Zentrum für Luft- und Raumfahrt (DLR)
Oberpfaffenhofen
Institut für Robotik und Mechatronik
Münchner Str. 20
82230 Wessling, Germany
e-mail: smagt@dlr.de

Gianmarco Veruggio

Consiglio Nazionale delle Ricerche
Istituto di Elettronica e di Ingegneria
dell'Informazione e delle Telecomunicazioni
Via De Marini 6
16149 Genova, Italy
e-mail: gianmarco@veruggio.it

Luigi Villani

Università degli Studi di Napoli Federico II
Dipartimento di Informatica e Sistemistica,
PRISMA Lab
Via Claudio 21
80125 Napoli, Italy
e-mail: luigi.villani@unina.it

Arto Visala

Helsinki University of Technology (TKK)
Department of Automation and Systems
Technology
Helsinki 02015, Finland
e-mail: arto.visala@tkk.fi

Kenneth Waldron

Stanford University
Department of Mechanical Engineering
Terman Engineering Center 521
Stanford, CA 94305-4021, USA
e-mail: kwaldron@stanford.edu

Ian D. Walker

Clemson University
Department of Electrical and Computer
Engineering
Clemson, SC 29634, USA
e-mail: ianw@ces.clemson.edu

Christian Wallraven

Max-Planck-Institut für biologische Kybernetik
Kognitive Humanpsychophysik
Spemannstr. 38
72076 Tübingen, Germany
e-mail: christian.wallraven@tuebingen.mpg.de

Brian Wilcox

California Institute of Technology
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109, USA
e-mail: brian.h.wilcox@jpl.nasa.gov

Jing Xiao

University of North Carolina
Department of Computer Science
Charlotte, NC 28223, USA
e-mail: xiao@uncc.edu

Dana R. Yoerger

Woods Hole Oceanographic Institution
Department of Applied Ocean Physics
and Engineering
MS7 Blake Bldg.
Woods Hole, MA 02543, USA
e-mail: dyoerger@whoi.edu

Kazuhito Yokoi

National Institute of Advanced Industrial Science
and Technology (AIST)
Intelligent Systems Research Institute
1-1-1 Umezono
305-8568 Tsukuba, Japan
e-mail: kazuhito.yokoi@aist.go.jp

Kazuya Yoshida

Tohoku University
Department of Aerospace Engineering
6-6-01 Aoba-yama
980-8579 Sendai, Japan
e-mail: yoshida@astro.mech.tohoku.ac.jp

Alexander Zelinsky

Commonwealth Scientific
and Industrial Research Organisation (CSIRO)
ICT Centre
Epping, Sydney NSW 1710, Australia
e-mail: alex.zelinsky@csiro.au

Contents

List of Abbreviations	LIII
Introduction	
<i>Bruno Siciliano, Oussama Khatib</i>	1
Part A Robotics Foundations	
1 Kinematics	
<i>Kenneth Waldron, James Schmiedeler</i>	9
1.1 Overview	9
1.2 Position and Orientation Representation	10
1.3 Joint Kinematics	18
1.4 Geometric Representation	23
1.5 Workspace	25
1.6 Forward Kinematics	26
1.7 Inverse Kinematics	27
1.8 Forward Instantaneous Kinematics	29
1.9 Inverse Instantaneous Kinematics	30
1.10 Static Wrench Transmission	30
1.11 Conclusions and Further Reading	31
References	31
2 Dynamics	
<i>Roy Featherstone, David E. Orin</i>	35
2.1 Overview	36
2.2 Spatial Vector Notation	37
2.3 Canonical Equations	43
2.4 Dynamic Models of Rigid-Body Systems	45
2.5 Kinematic Trees	50
2.6 Kinematic Loops	57
2.7 Conclusions and Further Reading	60
References	62
3 Mechanisms and Actuation	
<i>Victor Scheinman, J. Michael McCarthy</i>	67
3.1 Overview	67
3.2 System Features	68
3.3 Kinematics and Kinetics	69
3.4 Serial Robots	72
3.5 Parallel Robots	73
3.6 Mechanical Structure	75

3.7	Joint Mechanisms.....	76
3.8	Robot Performance.....	82
3.9	Conclusions and Further Reading	84
	References	84
4	Sensing and Estimation	
	<i>Henrik I. Christensen, Gregory D. Hager</i>	87
4.1	The Perception Process	88
4.2	Sensors	90
4.3	Estimation Processes	93
4.4	Representations	104
4.5	Conclusions and Further Readings	106
	References	106
5	Motion Planning	
	<i>Lydia E. Kavvaki, Steven M. LaValle</i>	109
5.1	Motion Planning Concepts	110
5.2	Sampling-Based Planning	111
5.3	Alternative Approaches	115
5.4	Differential Constraints	118
5.5	Extensions and Variations	121
5.6	Advanced Issues	124
5.7	Conclusions and Further Reading	127
	References	128
6	Motion Control	
	<i>Wankyun Chung, Li-Chen Fu, Su-Hau Hsu[†]</i>	133
6.1	Introduction to Motion Control.....	134
6.2	Joint Space Versus Operational Space Control	135
6.3	Independent-Joint Control	137
6.4	PID Control	139
6.5	Tracking Control	141
6.6	Computed-Torque Control.....	143
6.7	Adaptive Control	147
6.8	Optimal and Robust Control	150
6.9	Digital Implementation	153
6.10	Learning Control.....	155
	References	157
7	Force Control	
	<i>Luigi Villani, Joris De Schutter</i>	161
7.1	Background	161
7.2	Indirect Force Control	164
7.3	Interaction Tasks	171
7.4	Hybrid Force/Motion Control.....	177

7.5	Conclusions and Further Reading	181
	References	183
8	Robotic Systems Architectures and Programming	
	<i>David Kortenkamp, Reid Simmons</i>	187
8.1	Overview	187
8.2	History	189
8.3	Architectural Components	193
8.4	Case Study – GRACE	200
8.5	The Art of Robot Architectures	202
8.6	Conclusions and Further Reading	203
	References	204
9	AI Reasoning Methods for Robotics	
	<i>Joachim Hertzberg, Raja Chatila</i>	207
9.1	Knowledge Representation and Inference	208
9.2	KR Issues for Robots	212
9.3	Action Planning	214
9.4	Robot Learning	219
9.5	Conclusions and Further Reading	221
	References	222
 Part B Robot Structures		
10	Performance Evaluation and Design Criteria	
	<i>Jorge Angeles, Frank C. Park</i>	229
10.1	The Robot Design Process	229
10.2	Workspace Criteria	231
10.3	Dexterity Indices	235
10.4	Other Performance Indices	238
	References	242
11	Kinematically Redundant Manipulators	
	<i>Stefano Chiaverini, Giuseppe Oriolo, Ian D. Walker</i>	245
11.1	Overview	245
11.2	Task-Oriented Kinematics	247
11.3	Inverse Differential Kinematics	250
11.4	Redundancy Resolution via Optimization	255
11.5	Redundancy Resolution via Task Augmentation	256
11.6	Second-Order Redundancy Resolution	259
11.7	Cyclicity	260
11.8	Hyperredundant Manipulators	261
11.9	Conclusion and Further Reading	265
	References	265

12 Parallel Mechanisms and Robots	
<i>Jean-Pierre Merlet, Clément Gosselin</i>	269
12.1 Definitions	269
12.2 Type Synthesis of Parallel Mechanisms	271
12.3 Kinematics	271
12.4 Velocity and Accuracy Analysis	273
12.5 Singularity Analysis	274
12.6 Workspace Analysis	276
12.7 Static Analysis and Static Balancing	277
12.8 Dynamic Analysis	279
12.9 Design	279
12.10 Application Examples	280
12.11 Conclusion and Further Reading	281
References	281
13 Robots with Flexible Elements	
<i>Alessandro De Luca, Wayne Book</i>	287
13.1 Robots with Flexible Joints	288
13.2 Robots with Flexible Links	306
References	317
14 Model Identification	
<i>John Hollerbach, Wisama Khalil, Maxime Gautier</i>	321
14.1 Overview	321
14.2 Kinematic Calibration	323
14.3 Inertial Parameter Estimation	330
14.4 Identifiability and Numerical Conditioning	334
14.5 Conclusions and Further Reading	341
References	342
15 Robot Hands	
<i>Claudio Melchiorri, Makoto Kaneko</i>	345
15.1 Basic Concepts	346
15.2 Design of Robot Hands	347
15.3 Technologies for Actuation and Sensing	351
15.4 Modeling and Control of a Robot Hand	355
15.5 Applications and Trends	359
15.6 Conclusions and Further Reading	359
References	359
16 Legged Robots	
<i>Shuuji Kajita, Bernard Espiau</i>	361
16.1 A Brief History	362
16.2 Analysis of Cyclic Walking	363
16.3 Control of Biped Robots Using Forward Dynamics	366
16.4 Biped Robots in the ZMP Scheme	371
16.5 Multilegged Robots	378

16.6	Other Legged Robots	383
16.7	Performance Indices	385
16.8	Conclusions and Future Trends	386
	References	387
17	Wheeled Robots	
	<i>Guy Campion, Woojin Chung</i>	391
17.1	Overview	391
17.2	Mobility of Wheeled Robots	392
17.3	State-Space Models of Wheeled Mobile Robots	398
17.4	Structural Properties of Wheeled Robots Models	403
17.5	Wheeled Robot Structures	405
17.6	Conclusions	409
	References	410
18	Micro/Nanorobots	
	<i>Bradley J. Nelson, Lixin Dong, Fumihito Arai</i>	411
18.1	Overview of Micro- and Nanorobotics	411
18.2	Scaling	414
18.3	Actuation at the Micro- and Nanoscales	415
18.4	Sensing at the Micro- and Nanoscales	417
18.5	Fabrication	419
18.6	Microassembly	422
18.7	Microrobotics	427
18.8	Nanorobotics	431
18.9	Conclusions	443
	References	444
 Part C Sensing and Perception		
19	Force and Tactile Sensors	
	<i>Mark R. Cutkosky, Robert D. Howe, William R. Provancher</i>	455
19.1	Sensor Types	456
19.2	Tactile Information Processing	464
19.3	Integration Challenges	471
19.4	Conclusions and Future Developments	471
	References	471
20	Inertial Sensors, GPS, and Odometry	
	<i>Gregory Dudek, Michael Jenkin</i>	477
20.1	Odometry	477
20.2	Gyroscopic Systems	479
20.3	Accelerometers	482
20.4	IMU Packages	483
20.5	GPS	484
20.6	GPS-IMU Integration	488

20.7	Further Reading	489
20.8	Currently Available Hardware	489
	References	490
21	Sonar Sensing	
	<i>Lindsay Kleeman, Roman Kuc</i>	491
21.1	Sonar Principles	492
21.2	Sonar Beam Pattern	494
21.3	Speed of Sound	496
21.4	Waveforms	496
21.5	Transducer Technologies	497
21.6	Reflecting Object Models	499
21.7	Artifacts	500
21.8	TOF Ranging	501
21.9	Echo Waveform Coding	503
21.10	Echo Waveform Processing	506
21.11	CTFM Sonar	508
21.12	Multipulse Sonar	511
21.13	Sonar Rings	512
21.14	Motion Effects	513
21.15	Biomimetic Sonars	515
21.16	Conclusions	516
	References	517
22	Range Sensors	
	<i>Robert B. Fisher, Kurt Konolige</i>	521
22.1	Range Sensing Basics	521
22.2	Registration	530
22.3	Navigation and Terrain Classification	537
22.4	Conclusions and Further Reading	540
	References	540
23	3-D Vision and Recognition	
	<i>Kostas Daniilidis, Jan-Olof Eklundh</i>	543
23.1	3-D Vision and Visual SLAM	544
23.2	Recognition	551
23.3	Conclusion and Further Reading	558
	References	559
24	Visual Servoing and Visual Tracking	
	<i>François Chaumette, Seth Hutchinson</i>	563
24.1	The Basic Components of Visual Servoing	564
24.2	Image-Based Visual Servo	565
24.3	Position-Based Visual Servo	572
24.4	Advanced Approaches	574
24.5	Performance Optimization and Planning	577
24.6	Estimation of 3-D Parameters	578

24.7	Target Tracking.....	579
24.8	Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space.....	580
24.9	Conclusions.....	581
	References	582

25 Multisensor Data Fusion

	<i>Hugh Durrant-Whyte, Thomas C. Henderson</i>	585
25.1	Multisensor Data Fusion Methods.....	585
25.2	Multisensor Fusion Architectures.....	598
25.3	Applications.....	603
25.4	Conclusions.....	608
	References	608

Part D Manipulation and Interfaces

26 Motion for Manipulation Tasks

	<i>Oliver Brock, James Kuffner, Jing Xiao</i>	615
26.1	Overview.....	615
26.2	Task-Level Control.....	618
26.3	Manipulation Planning.....	622
26.4	Assembly Motion.....	628
26.5	Unifying Feedback Control and Planning.....	634
26.6	Conclusions and Further Reading.....	637
	References	639

27 Contact Modeling and Manipulation

	<i>Imin Kao, Kevin Lynch, Joel W. Burdick</i>	647
27.1	Overview.....	647
27.2	Kinematics of Rigid-Body Contact.....	648
27.3	Forces and Friction.....	652
27.4	Rigid-Body Mechanics with Friction.....	655
27.5	Pushing Manipulation.....	658
27.6	Contact Interfaces and Modeling.....	659
27.7	Friction Limit Surface.....	661
27.8	Contacts in Grasping and Fixture Designs.....	664
27.9	Conclusions and Further Reading.....	666
	References	666

28 Grasping

	<i>Domenico Prattichizzo, Jeffrey C. Trinkle</i>	671
28.1	Background.....	671
28.2	Models and Definitions.....	672
28.3	Controllable Twists and Wrenches.....	677
28.4	Restraint Analysis.....	680
28.5	Examples.....	687

28.6	Conclusion and Further Reading	697
	References	698
29	Cooperative Manipulators	
	<i>Fabrizio Caccavale, Masaru Uchiyama</i>	701
29.1	A Historical Overview	701
29.2	Kinematics and Statics	703
29.3	Cooperative Task Space	707
29.4	Dynamics and Load Distribution	708
29.5	Task-Space Analysis	710
29.6	Control	711
29.7	Conclusions and Further Reading	715
	References	716
30	Haptics	
	<i>Blake Hannaford, Allison M. Okamura</i>	719
30.1	Overview	719
30.2	Haptic Device Design	724
30.3	Haptic Rendering	727
30.4	Control and Stability of Haptic Interfaces	730
30.5	Tactile Display	731
30.6	Conclusions and Further Reading	735
	References	735
31	Telerobotics	
	<i>Günter Niemeyer, Carsten Preusche, Gerd Hirzinger</i>	741
31.1	Overview	741
31.2	Telerobotic Systems and Applications	743
31.3	Control Architectures	746
31.4	Bilateral Control and Force Feedback	751
31.5	Conclusions and Further Reading	754
	References	755
32	Networked Telerobots	
	<i>Dezhen Song, Ken Goldberg, Nak Young Chong</i>	759
32.1	Overview and Background	759
32.2	A Brief History	760
32.3	Communications and Networking	761
32.4	Conclusion and Future Directions	769
	References	769
33	Exoskeletons for Human Performance Augmentation	
	<i>Homayoon Kazerooni</i>	773
33.1	Survey of Exoskeleton Systems	773
33.2	Upper-Extremity Exoskeleton	775
33.3	Intelligent Assist Device	776
33.4	Control Architecture for Upper-Extremity Exoskeleton Augmentation	778

33.5 Applications of Intelligent Assist Device	780
33.6 Lower-Extremity Exoskeleton	780
33.7 The Control Scheme of an Exoskeleton	782
33.8 Highlights of the Lower-Extremity Design	786
33.9 Field-Ready Exoskeleton Systems	790
33.10 Conclusion and Further Reading	792
References	792

Part E Mobile and Distributed Robotics

34 Motion Control of Wheeled Mobile Robots

<i>Pascal Morin, Claude Samson</i>	799
34.1 Background	800
34.2 Control Models	801
34.3 Adaptation of Control Methods for Holonomic Systems	804
34.4 Methods Specific to Nonholonomic Systems	806
34.5 Complementary Issues and Bibliographical Guide	823
References	825

35 Motion Planning and Obstacle Avoidance

<i>Javier Minguez, Florent Lamiraux, Jean-Paul Laumond</i>	827
35.1 Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory	828
35.2 Kinematic Constraints and Controllability	829
35.3 Motion Planning and Small-Time Controllability	830
35.4 Local Steering Methods and Small-Time Controllability	832
35.5 Robots and Trailers	835
35.6 Approximate Methods	837
35.7 From Motion Planning to Obstacle Avoidance	837
35.8 Definition of Obstacle Avoidance	838
35.9 Obstacle Avoidance Techniques	839
35.10 Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance	845
35.11 Integration Planning – Reaction	847
35.12 Conclusions, Future Directions, and Further Reading	849
References	850

36 World Modeling

<i>Wolfram Burgard, Martial Hebert</i>	853
36.1 Historical Overview	854
36.2 World Models for Indoors and Structured Environments	855
36.3 World and Terrain Models for Natural Environments	859
36.4 Dynamic Environments	866
References	867

37 Simultaneous Localization and Mapping

<i>Sebastian Thrun, John J. Leonard</i>	871
37.1 Overview	871

37.2	SLAM: Problem Definition.....	872
37.3	The Three Main SLAM Paradigms.....	875
37.4	Conclusion and Future Challenges.....	885
37.5	Suggestions for Further Reading.....	886
	References	886
38	Behavior-Based Systems	
	<i>Maja J. Matarić, François Michaud</i>	891
38.1	Robot Control Approaches.....	891
38.2	Basic Principles of Behavior-Based Systems.....	894
38.3	Basis Behaviors	897
38.4	Representation in Behavior-Based Systems.....	897
38.5	Learning in Behavior-Based Systems.....	898
38.6	Continuing Work	902
38.7	Conclusions and Further Reading	905
	References	906
39	Distributed and Cellular Robots	
	<i>Zack Butler, Alfred Rizzi</i>	911
39.1	Modularity for Locomotion.....	911
39.2	Modularity for Manipulation	914
39.3	Modularity for Geometric Reconfiguration of Robot Systems	915
39.4	Modularity for Robustness	918
39.5	Conclusions and Further Reading	918
	References	919
40	Multiple Mobile Robot Systems	
	<i>Lynne E. Parker</i>	921
40.1	History	922
40.2	Architectures for Multirobot Systems	922
40.3	Communication.....	925
40.4	Swarm Robots	926
40.5	Heterogeneity	928
40.6	Task Allocation	930
40.7	Learning	932
40.8	Applications.....	933
40.9	Conclusions and Further Reading	935
	References	936
41	Networked Robots	
	<i>Vijay Kumar, Daniela Rus, Gaurav S. Sukhatme</i>	943
41.1	Overview.....	943
41.2	State of the Art and Potential	945
41.3	Research Challenges	947
41.4	Control	949
41.5	Communication for Control	950
41.6	Communication for Perception.....	951

41.7 Control for Perception	952
41.8 Control for Communication	953
41.9 Conclusions and Further Reading	955
References	955

Part F Field and Service Robotics

42 Industrial Robotics

<i>Martin Hägele, Klas Nilsson, J. Norberto Pires</i>	963
42.1 A Short History of Industrial Robots	964
42.2 Typical Applications and Robot Configurations	969
42.3 Kinematics and Mechanisms	975
42.4 Task Descriptions – Teaching and Programming	976
42.5 End-Effectors and System Integration	980
42.6 Conclusions and Long-Term Challenges	983
References	985

43 Underwater Robotics

<i>Gianluca Antonelli, Thor I. Fossen, Dana R. Yoerger</i>	987
43.1 The Expanding Role of Marine Robotics in Oceanic Engineering	987
43.2 Underwater Robotics	989
43.3 Applications	1003
43.4 Conclusions and Further Reading	1005
References	1005

44 Aerial Robotics

<i>Eric Feron, Eric N. Johnson</i>	1009
44.1 Background	1010
44.2 History of Aerial Robotics	1010
44.3 Applications of Aerial Robotics	1012
44.4 Current Challenges	1014
44.5 Basic Aerial Robot Flight Concepts	1015
44.6 The Entry Level for Aerial Robotics: Inner-Loop Control	1020
44.7 Active Research Areas	1022
44.8 Conclusions and Further Reading	1026
References	1027

45 Space Robots and Systems

<i>Kazuya Yoshida, Brian Wilcox</i>	1031
45.1 Historical Developments and Advances of Orbital Robotic Systems ...	1032
45.2 Historical Developments and Advances of Surface Robotic Systems ..	1037
45.3 Mathematical Modeling	1044
45.4 Future Directions of Orbital and Surface Robotic Systems	1056
45.5 Conclusions and Further Reading	1060
References	1060

46 Robotics in Agriculture and Forestry	
<i>John Billingsley, Arto Visala, Mark Dunn</i>	1065
46.1 Definitions	1066
46.2 Forestry	1066
46.3 Broad Acre Applications	1070
46.4 Horticulture	1071
46.5 Livestock	1072
46.6 Unmanned Vehicles	1075
46.7 Conclusions and Future Directions	1075
References	1075
47 Robotics in Construction	
<i>Kamel S. Saidi, Jonathan B. O'Brien, Alan M. Lytle</i>	1079
47.1 Overview	1080
47.2 Economic Aspects	1085
47.3 Applications	1086
47.4 Currently Unsolved Technical Problems	1093
47.5 Future Directions	1095
47.6 Conclusions and Further Reading	1096
References	1096
48 Robotics in Hazardous Applications	
<i>James P. Trevelyan, Sung-Chul Kang, William R. Hamel</i>	1101
48.1 Operation in Hazardous Environments: The Need for a Robotics Solution	1101
48.2 Applications	1103
48.3 Enabling Technologies	1114
48.4 Conclusions and Further Reading	1122
References	1123
49 Mining Robotics	
<i>Peter Corke, Jonathan Roberts, Jock Cunningham, David Hainsworth</i>	1127
49.1 Background	1127
49.2 Metalliferous Mining	1132
49.3 Underground Coal Mining	1140
49.4 Surface Coal Mining	1144
49.5 Conclusions and Further Reading	1148
References	1148
50 Search and Rescue Robotics	
<i>Robin R. Murphy, Satoshi Tadokoro, Daniele Nardi, Adam Jacoff, Paolo Fiorini, Howie Choset, Aydan M. Erkmén</i>	1151
50.1 Overview	1152
50.2 Disaster Characteristics and Impact on Robots	1153
50.3 Robots Actually Used at Disasters	1156
50.4 Promising Robots and Concepts	1161
50.5 Evaluation and Benchmarks	1165

50.6	Fundamental Problems and Open Issues	1167
50.7	Conclusions and Further Reading	1171
	References	1171
51	Intelligent Vehicles	
	<i>Alberto Broggi, Alexander Zelinsky, Michel Parent, Charles E. Thorpe</i>	1175
51.1	Why Intelligent Vehicles?	1175
51.2	Enabling Technologies	1178
51.3	Road Scene Understanding	1181
51.4	Advanced Driver Assistance	1184
51.5	Driver Monitoring	1188
51.6	Automated Vehicles	1191
51.7	Future Trends and Prospects	1194
51.8	Conclusions and Further Reading	1195
	References	1195
52	Medical Robotics and Computer-Integrated Surgery	
	<i>Russell H. Taylor, Arianna Menciassi, Gabor Fichtinger, Paolo Dario</i>	1199
52.1	Core Concepts	1200
52.2	Technology	1204
52.3	Systems, Research Areas, and Applications	1209
52.4	Conclusion and Future Directions	1217
	References	1218
53	Rehabilitation and Health Care Robotics	
	<i>H.F. Machiel Van der Loos, David J. Reinkensmeyer</i>	1223
53.1	Overview	1223
53.2	Physical Therapy and Training Robots	1227
53.3	Aids for People with Disabilities	1235
53.4	Smart Prostheses and Orthoses	1240
53.5	Augmentation for Diagnosis and Monitoring	1242
53.6	Safety, Ethics, Access, and Economics	1244
53.7	Conclusions and Further Readings	1245
	References	1246
54	Domestic Robotics	
	<i>Erwin Prassler, Kazuhiro Kosuge</i>	1253
54.1	Cleaning Robots	1254
54.2	Lawn-Mowing Robots	1271
54.3	Smart Appliances	1273
54.4	Smart Homes	1275
54.5	Domestic Robotics: It Is the Business Case Which Matters	1279
54.6	Conclusions and Further Reading	1280
	References	1280

55 Robots for Education

<i>David P. Miller, Illah R. Nourbakhsh, Roland Siegwart</i>	1283
55.1 The Role of Robots in Education	1284
55.2 Educational Robot Tournaments.....	1285
55.3 Education Robot Platforms.....	1287
55.4 Education Robot Controllers and Programming Environments.....	1290
55.5 Robots and Informal Learning Venues (Museums)	1292
55.6 Educational Evaluation of Robot Programs	1296
55.7 Conclusions and Further Reading	1298
References	1298

Part G Human-Centered and Life-Like Robotics**56 Humanoids**

<i>Charles C. Kemp, Paul Fitzpatrick, Hirohisa Hirukawa, Kazuhito Yokoi, Kensuke Harada, Yoshio Matsumoto</i>	1307
56.1 Why Humanoids?	1307
56.2 History and Overview	1310
56.3 Locomotion	1312
56.4 Manipulation	1315
56.5 Whole-Body Activities	1318
56.6 Communication	1325
56.7 Conclusions and Further Reading	1329
References	1329

57 Safety for Physical Human-Robot Interaction

<i>Antonio Bicchi, Michael A. Peshkin, J. Edward Colgate</i>	1335
57.1 Motivations for Safe pHRI	1336
57.2 Safety for Hands-Off pHRI	1337
57.3 Design of Intrinsically Safe Robots	1338
57.4 Safety for Hands-On pHRI	1341
57.5 Safety Standards for pHRI	1345
57.6 Conclusions.....	1346
References	1346

58 Social Robots that Interact with People

<i>Cynthia Breazeal, Atsuo Takanishi, Tetsunori Kobayashi</i>	1349
58.1 Social Robot Embodiment	1350
58.2 Multimodal Communication	1352
58.3 Expressive Emotion-Based Interaction	1356
58.4 Socio-cognitive Skills	1360
58.5 Conclusion and Further Reading	1365
References	1366

59 Robot Programming by Demonstration

<i>Aude Billard, Sylvain Calinon, Rüdiger Dillmann, Stefan Schaal</i>	1371
59.1 History	1372

59.2	Engineering-Oriented Approaches.....	1374
59.3	Biologically-Oriented Learning Approaches	1386
59.4	Conclusions and Open Issues in Robot PbD	1389
	References	1389
60	Biologically Inspired Robots	
	<i>Jean-Arcady Meyer, Agnès Guillot</i>	1395
60.1	General Background	1395
60.2	Bio-inspired Morphologies	1396
60.3	Bio-inspired Sensors	1398
60.4	Bio-inspired Actuators	1402
60.5	Bio-inspired Control Architectures	1408
60.6	Energetic Autonomy	1412
60.7	Collective Robotics.....	1413
60.8	Biohybrid Robots.....	1415
60.9	Discussion	1417
60.10	Conclusion	1418
	References	1418
61	Evolutionary Robotics	
	<i>Dario Floreano, Phil Husbands, Stefano Nolfi</i>	1423
61.1	Method	1423
61.2	First Steps	1424
61.3	Simulation and Reality	1428
61.4	Simple Controllers, Complex Behaviors	1429
61.5	Seeing the Light	1431
61.6	Computational Neuroethology	1434
61.7	Evolution and Learning	1439
61.8	Competition and Cooperation	1442
61.9	Evolutionary Hardware	1444
61.10	Closing Remarks	1447
	References	1447
62	Neurorobotics: From Vision to Action	
	<i>Michael A. Arbib, Giorgio Metta, Patrick van der Smagt</i>	1453
62.1	Definitions	1453
62.2	Neuroethological Inspiration	1454
62.3	The Role of the Cerebellum	1462
62.4	The Role of Mirror Systems	1467
62.5	Extroduction	1474
62.6	Further Reading	1475
	References	1475
63	Perceptual Robotics	
	<i>Heinrich H. Bülthoff, Christian Wallraven, Martin A. Giese</i>	1481
63.1	Overview.....	1481
63.2	Example-Based Object Representations.....	1483

63.3	Example-Based Movement Representations	1490
63.4	Example-Based Synthesis Models: From Faces to Movements	1492
63.5	Conclusions and Further Reading	1494
	References	1495
64	Roboethics: Social and Ethical Implications of Robotics	
	<i>Gianmarco Veruggio, Fiorella Operto</i>	1499
64.1	A Methodological Note	1501
64.2	Specificity of Robotics	1502
64.3	What Is a Robot?	1502
64.4	Cultural Differences in Robot's Acceptance	1503
64.5	From Literature to Today's Debate	1503
64.6	Roboethics	1504
64.7	Ethics and Morality	1505
64.8	Moral Theories	1505
64.9	Ethics in Science and Technology	1506
64.10	Conditions for Implementation	1507
64.11	Operativeness of the Principles	1507
64.12	Ethical Issues in an ICT Society	1507
64.13	Harmonization of Principles	1509
64.14	Ethics and Professional Responsibility	1510
64.15	Roboethics Taxonomy	1511
64.16	Conclusions and Further Reading	1519
	References	1522
	Acknowledgements	1525
	About the Authors	1527
	Detailed Contents	1555
	Subject Index	1591

List of Abbreviations

A		B	
AAAI	American Association for Artificial Intelligence	BIOROB	biomimetic robotics
ABA	articulated-body algorithm	BLDC	brushless direct current
ABRT	automated bus rapid transit	BLE	broadcast of local eligibility
ACAS	airborne collision avoidance systems	BLEEX	Berkeley lower-extremity exoskeleton
ACC	adaptive cruise control	BLUE	best linear unbiased estimator
ACM	active cord mechanism	BN	Bayes network
ACM	Association of Computing Machinery	BRT	bus rapid transit
ADAS	advanced driver assistance systems	C	
ADL	activities of daily living	C/A	coarse-acquisition
ADSL	asymmetric digital subscriber line	CAM	computer-aided manufacturing
AGV	automated guided vehicles	CAD	computer-aided design
AHS	advanced highway systems	CAE	computer-aided engineering
AI	artificial intelligence	CALM	continuous air interface long and medium range
AIP	anterior interparietal area	CAN	controller area network
AIS	artificial intelligence (AI) system	CARD	computer-aided remote driving
AISB	artificial intelligence and simulation o behavior	CASPER	continuous activity scheduling, planning, execution and replanning
AIT	anterior inferotemporal cortex	CAT	computer-aided tomography
AM	actuators for manipulation	CB	cluster bombs
AMA	artificial moral agents	CCD	charge-coupled devices
AMD	autonomous mental development	CCI	control command interpreter
ANSI	American National Standards Institute	CCP	coverage configuration protocol
AP	antipersonnel	CCT	conservative congruence transformation
APG	adjustable pattern generator	CCW	counterclockwise
AR	augmented reality	CE	computer ethics
ARAMIS	Space Application of Automation, Robotics and Machine Intelligence	CEA	Commission de Energie Atomique
ASCL	adaptive seek control logic	CEBOT	cellular robot
ASD	autism spectrum disorder	CF	climbing fibers
ASIC	application-specific integrated circuit	CF	contact formation
ASKA	receptionist robot	CG	center of gravity
ASM	advanced servomanipulator	CGA	clinical gait analysis
ASN	active sensor network	CGI	common gateway interface
ASTRO	autonomous space transport robotic operations	CIE	International Commission on Illumination
ASV	adaptive suspension vehicle	CIRCA	cooperative intelligent real-time control architecture
AT	antitank	CIS	computer-integrated surgery
ATLSS	advanced technology for large structural systems	CLARAty	coupled layered architecture for robot autonomy
ATR	Advanced Telecommunications Research Institute International	CLEaR	closed-loop execution and recovery
AuRA	autonomous robot architecture	CLIK	closed-loop inverse kinematics
AUV	autonomous underwater vehicles	CMAC	cerebellar model articulation controller
AV	antivehicle	CML	concurrent mapping and localization
		CNC	computer numerical control
		CNP	contract net protocol

CNT carbon nanotubes
 COG center of gravity
 CONE Collaborative Observatory for Nature
 Environments

CONRO configurable robot
 COR center of rotation
 CORBA common object request broker
 architecture
 COV characteristic output vector
 CP closest point
 CP complementarity problem
 CP cerebral palsy
 CPG central pattern generators
 CPSR computer professional for social
 responsibility
 CRBA composite-rigid-body algorithm
 CRLB Cramer–Rao lower bound
 CSIRO Commonwealth Scientific and Industrial
 Research Organization
 CSMA carrier sense multiple access
 CT computed tomography
 CTFM continuous-transmission
 frequency-modulated
 CTL cut-to-length
 CU control unit
 CVIS cooperative vehicle infrastructure systems
 CW clockwise

D

DARPA Defense Advanced Research Projects
 Agency
 DARS distributed autonomous robotics systems
 DBNs dynamic Bayesian networks
 DD differentially driven
 DDF decentralized data fusion
 DeVAR desktop vocational assistant robot
 DFRA distributed field robot architecture
 DFT discrete Fourier transform
 DGA Delegation Generale pour L'Armement
 DH Denavit–Hartenberg
 DIO digital input-output
 DIRA distributed robot architecture
 DL description logics
 DLR Deutsches Zentrum für Luft- und
 Raumfahrt
 DM2 distributed macro-mini actuation
 DoD Department of Defense
 DOF degree of freedom
 DOG difference of Gaussian
 DOP dilution of precision
 DPN dip-pen nanolithography
 DRIE deep reactive ion etching
 DSM dynamic state machine
 DSO Defense Sciences Office
 DSRC dedicated short-range communications

DVL Doppler velocity log
 DWA dynamic window approach

E

EBA extratriate body part area
 EBID electron-beam-induced deposition
 ECU electronics controller unit
 EDM electrical discharge machining
 EDM electronic distance measuring
 EEG electroencephalogram
 EGNOS Euro Geostationary Navigation Overlay
 Service
 EKF extended Kalman filter
 EM expectation maximization
 EMG electromyography
 EMS electrical master–slave manipulators
 ENSICA Ecole Nationale Supérieure des
 Constructions Aeronautiques
 EO elementary operators
 EOD explosive ordnance disposal
 EP exploratory procedures
 EPFL Ecole Polytechnique Fédérale de
 Lausanne
 EPP extended physiological proprioception
 ERA European robotic arm
 ES electrical stimulation
 ESA European Space Agency
 ESL execution support language
 ETS engineering test satellite
 EVA extravehicular activity

F

FARS Fagg-Arbib-Rizzolatti-Sakata
 FE finite element
 FESEM field-emission SEM
 FIFO first-in first-out
 fMRI functional magnetic resonance imaging
 FMS flexible manufacturing systems
 FNS functional neural stimulation
 FOPL first-order predicate logic
 FPGAs field programmable gate array
 FRI foot rotating indicator
 FSA finite-state acceptors
 FSM finite-state machine
 FSR force sensing resistor
 FST finite-state transducer
 FSW feasible solution of wrench
 FTTH fiber to the home

G

GAS global asymptotic stability
 GBAS ground-based augmentation systems
 GCR goal-contact relaxation

GDP	gross domestic product	IA	instantaneous allocation
GenoM	generator of modules	IAD	intelligent assist device
GEO	geostationary Earth orbit	ICA	independent component analysis
GI	gastrointestinal	ICBL	International Campaign to Ban Landmines
GICHD	Geneva International Center for Humanitarian Demining	ICE	internet communications engine
GJM	generalized Jacobian matrix	ICP	iterative closest-point algorithm
GLS	Global Navigation Satellite System Landing System	ICR	instantaneous center of rotation
GMM	Gaussian mixture model	ICRA	International Conference on Robotics and Automation
GMR	Gaussian mixture regression	ICT	information and communication technology
GNS	global navigation systems	IDL	interface definition language
GNSS	global navigation satellite system	IE	information ethics
GP	Gaussian processes	IED	improvised explosive device
GPR	ground-penetrating radar	IEEE	Institute of Electrical and Electronics Engineers
GPRS	general packet radio service	IETF	Internet engineering task force
GPS	global positioning system	IFRR	International Foundation of Robotics Research
GRACE	graduate robot attending conference	iGPS	indoor GPS
GSD	geon structural description	IHIP	intermediate haptic interaction points
GSI	Gadd's severity index	IK	inverse kinematics
GUI	graphical user interface	ILP	inductive logic programming
GZMP	generalized ZMP	ILS	instrument landing system
H		IMTS	intelligent multimode transit system
HAL	hybrid assisted limb	IMU	inertial measurement units
HAMMER	hierarchical attentive multiple models for execution and recognition	IOSS	input-output-to-state stability
HCI	human computer interaction	IP	internet protocol
HD	haptic device	IPC	interprocess communication
HDSL	high data rate digital subscriber line	ISO	International Organization for Standardization
HEPA	semi-high efficiency-particulate airfilter	ISP	internet service provider
HF	hard-finger	ISS	input-to-state stability
HIC	head injury criterion	IST	Information Society Technologies
HIP	haptic interaction point	IST	Instituto Superior Técnico
HJB	Hamilton–Jacobi–Bellman	IT	intrinsic tactile
HJI	Hamilton–Jacobi–Isaac	IT	inferotemporal
HMD	head-mounted display	ITD	interaural time difference
HMM	hidden Markov model	IxTeT	indexed time table
HMX	high melting point explosives	J	
HO	human operator	JAUS	joint architecture for unmanned systems
HRI	human–robot interaction	JAXA	Japan space exploration agency
HRTEM	high-resolution transmission electron microscopes	JDL	joint directors of the laboratories
HST	Hubble space telescope	JEMRMS	Japanese experiment module remote manipulator system
HSTAMIDS	handheld standoff mine detection system	JHU	Johns Hopkins University
HTML	hypertext markup language	JND	just noticeable difference
HTN	hierarchical task network	JPL	Jet Propulsion Laboratory
I		JSIM	joint-space inertia matrix
I/O	input/output	JSP	Java Server Pages
I3CON	industrialized, integrated, intelligent, construction		

K

KR knowledge representation

L

LAAS Laboratoire d'Analyse et d'Architecture des Systèmes
 LADAR laser radar or laser detection and ranging
 LAN local-area network
 LARC Lie algebra rank condition
 LBL long-baseline system
 LCSP linear constraint satisfaction program
 LGN lateral geniculate nucleus
 LIDAR light detection and ranging
 LOS line of sight
 LP linear program
 LQG linear quadratic Gaussian
 LSS logical sensor system
 LVDT linear variable differential transformer
 LWR locally weighted regression

M

MACA Afghanistan Mine Action Center
 MANET mobile ad hoc network
 MAP maximum a posteriori probability
 MBARI Monterey Bay Aquarium Research Institute
 MBE molecular-beam epitaxy
 MBS mobile base system
 MC Monte Carlo
 MCS mission control system
 MDP Markovian decision process
 MST microsystem technology
 MEMS microelectromechanical systems
 MER Mars exploration rovers
 MESUR Mars environmental survey
 MF Mossy fibers
 MIA mechanical impedance adjuster
 MIG metal inert gas
 MIMO multi-input multi-output
 MIR mode identification and recovery
 MIS minimally invasive surgery
 MITI Ministry of International Trade and Industry
 ML maximum likelihood
 ML machine learning
 MLE maximum-likelihood estimation
 MLS multilevel surface map
 MNS mirror neuron system
 MOCVD metallo-organic chemical vapor deposition
 MOMR multiple operator multiple robot
 MOSR multiple operator single robot
 MPC model predictive control

MPFIM multiple paired forward-inverse models
 MPM manipulator positioning mechanism
 MR multirobot tasks
 MR multiple reflection
 MR magnetorheological
 MRAC model reference adaptive control
 MRI magnetic resonance imaging
 MRL manipulator retention latch
 MRSR Mars rover sample return
 MRTA multirobot task allocation
 MSAS Multifunctional Satellite Augmentation System
 MSER maximally stable extremal regions
 MSM master-slave manipulator

N

MT multitask
 MT medial temporal
 MTBF mean time between failure
 MTRAN modular transformer
 NAP nonaccidental properties
 NASA National Aeronautics and Space Agency
 NASDA National Space Development Agency of Japan
 NASREM NASA/NBS standard reference model
 NBS National Bureau of Standards
 NCEA National Center for Engineering in Agriculture
 NCER National Conference on Educational Robotics
 ND nearness diagram navigation
 NDDS network data distribution service
 NEMO network mobility
 NEMS nanoelectromechanical systems
 NICT National Institute of Information and Communications Technology
 NIDRR National Institute on Disability and Rehabilitation Research
 NIMS networked infomechanical systems
 NIOSH National Institute for Occupational Health and Safety
 NMEA National Marine Electronics Association
 NN neural networks
 NPS Naval Postgraduate School
 NRM nanorobotic manipulators
 NURBS non-uniform rational B-spline

O

OASIS onboard autonomous science investigation system
 OBSS orbiter boom sensor system
 OCU operator control units
 ODE ordinary differential equation

OH&S	occupation health and safety
OLP	offline programming
OM	optical microscope
ORB	object request brokers
ORCCAD	open robot controller computer aided design
ORM	obstacle restriction method
ORU	orbital replacement unit
OSIM	operational-space inertia matrix

P

P&O	prosthetics and orthotics
PAPA	privacy, accuracy, intellectual property, and access
PAS	pseudo-amplitude scan
PB	parametric bias
PbD	programming by demonstration
PC	principal contact
PC	Purkinje cells
PCA	principle components analysis
PD	proportional-derivative
PDDL	planning domain description language
PDGF	power data grapple fixtures
PEAS	probing environment and adaptive sleeping protocol
PET	positron emission tomography
PF	parallel fibers
PFC	prefrontal cortex
PFM	potential field method
pHRI	physical human–robot interaction
PI	policy iteration
PIC	programmable interrupt controller
PIC	programmable intelligent computer
PID	proportional–integral–derivative
PIT	posterior inferotemporal cortex
PKM	parallel kinematic machine
PLC	programmable logic controller
PLD	programmable logic device
PLEXIL	plan execution interchange language
PMD	photonic mixer device
PMMA	polymethyl methacrylate
PNT	Petri net transducers
POMDP	partially observable MDP
PPRK	palm pilot robot kit
PPS	precise positioning system
PR	photoresist
PRISMA	Projects of Robotics for Industry and Services, Mechatronics and Automation
PRM	probabilistic roadmap method
PRN	pseudorandom noise
PRS	procedural reasoning system
PS	power source
PTP	point-to-point

PTU	pan–tilt unit
PVDF	polyvinylidene fluoride
PwoF	point-contact-without-friction
PZT	lead zirconate titanate

Q

QD	quantum dot
QRIO	quest for curiosity
QT	quasistatic telerobotics

R

R.U.R.	Rossum’s Universal Robots
RAIM	receiver autonomous integrity monitoring
RALPH	rapidly adapting lane position handler
RAM	random-access (volatile) memory
RANSAC	random sample consensus
RAP	reactive action packages
RAS	Robotics and Automation Society
RBC	recognition-by-components
RBF	radial basis function
RC	radio-controlled
RCC	remote center of compliance
RCM	remote center of motion
RCR	responsible conduct of research
RCS	real-time control system
RERC	Rehabilitation Engineering Research Center on Rehabilitation Robotics
RF	radiofrequency
RFID	radiofrequency identification
RFWR	receptive field weighted regression
RG	rate gyros
RGB	red, green, blue
RIG	rate-integrating gyros
RL	reinforcement learning
RLG	random loop generator
RMMS	reconfigurable modular manipulator system
RNEA	recursive Newton–Euler algorithm
RNNPB	recurrent neural network with parametric bias
RNS	reaction null space
ROC	receiver operating curve
ROKVISS	robotic components verification on the ISS
ROM	read-only memory
ROTEX	robot technology experiment
ROV	remotely operated vehicle
RPC	remote procedure call
RPI	Rensselaer Polytechnic Institute
RPV	remotely piloted vehicle
RRT	rapid random tree
RSS	realistic robot simulation

RT reaction time
 RT room-temperature
 RTCA Radio Technical Commission for
 Aeronautics
 RTD resistance temperature device
 RTI real-time innovations
 RTK real-time kinematics
 RTS real-time system
 RWI real-world interface
 RWS robotic work station

S

SA selective availability
 SAIC Science Applications International, Inc.
 SAIL Stanford Artificial Intelligence
 Laboratory
 SAN semiautonomous navigation
 SBAS satellite-based augmentation systems
 SBL short-baseline system
 SCARA selective compliance assembly robot arm
 SCI spinal cord injury
 SDK standard development kit
 SDR software for distributed robotics
 SDV spatial dynamic voting
 SEA series elastic actuator
 SEE standard end-effector
 SELF sensorized environment for life
 SEM scanning electron microscopes
 SET single-electron transistors
 SF soft-finger
 SfM structure from motion
 SFX sensor fusion effects
 SGAS semiglobal asymptotic stability
 SGUUB semiglobal uniform ultimate boundedness
 SHOP simple hierarchical ordered planner
 SIFT scale-invariant feature transformation
 SIGMOD Special Interest Group on Management of
 Data
 SIPE system for interactive planning and
 execution monitoring
 SIR sampling importance resampling
 SISO single-input single-output
 SKM serial kinematic machines
 SLAM simultaneous localization and mapping
 SLAMP sheep loading animal manipulation
 platform
 SLICE specification language for ICE
 SLRV surveyor lunar rover vehicle
 SMA shape-memory alloy
 SMC sequential Monte Carlo
 SNOM scanning near-field OM
 SOI silicon-on-insulator
 SOMR single operator multiple robot
 SOSR single operator single robot
 SPA sense-plan-act

SPDM special-purpose dexterous manipulator
 SPS standard position system
 SR single-robot
 SRMS shuttle remote manipulator system
 SSRMS Space shuttle remote manipulator
 system
 ST single-task
 STM scanning tunneling microscopes
 STS superior temporal sulcus
 SVD singular value decomposition
 SWNT single-walled carbon nanotubes

T

TA time-extended assignment
 TAP test action pairs
 TC technical committee
 TCP transmission control protocol
 TDL task description language
 TDT tension differential type
 TEM transmission electron microscopes
 tEODor telerob explosive ordnance disposal and
 observation robot
 TF-IDF term-frequency inverse document
 frequency
 TMS transcranial magnetic stimulation
 TOF time of flight
 TPBVP two-point boundary value problem
 TSEE teleoperated small emplacement
 excavator
 TSP telesensor programming
 TTI thoracic trauma index
 TTS text-to-speech

U

UAS unmanned aerial systems
 UAV unmanned aerial vehicles
 UDP user data protocol
 UGV unmanned ground vehicle
 UML unified modeling language
 URL uniform resource locator
 US ultrasound
 USBL ultrashort-baseline system
 USV unmanned surface vehicle
 UUB uniform ultimate boundedness
 UUV unmanned underwater vehicles
 UVMS underwater vehicle manipulator system
 UWB ultra-wideband
 UXO unexploded ordnance

V

VANET vehicular ad-hoc network
 VC viscous injury response
 VCR videocassette recorder

vdW	van der Waals
VFH	vector field histogram
VI	value iteration
VIA	variable-impedance actuation
VLSI	very-large-scale integrated
VM	virtual manipulator
VO	velocity obstacles
VOR	vestibular-ocular reflex
VOR	VHF omnidirectional range
VR	virtual reality
VRML	virtual reality modeling language
VVV	versatile volumetric vision

W

WABIAN	Waseda bipedal humanoid
WAM	whole-arm manipulator

WAN	wide-area network
WG	world graph
WMR	wheeled mobile robot
WMSD	work-related musculoskeletal disorders
WTA	winner-take-all
WWW	world wide web

X

XHTML	extensible hyper text markup language
XML	extensible markup language

Z

ZMP	zero-moment point
ZP	zona pellucida

Introduction

Bruno Siciliano, Oussama Khatib

Robots! Robots on Mars and in oceans, in hospitals and homes, in factories and schools; robots fighting fires, making goods and products, saving time and lives . . . Robots today are making a considerable impact on many aspects of modern life, from industrial manufacturing to healthcare, transportation, and exploration of the deep space and sea. Tomorrow, robots will be as pervasive and personal as today's personal computers. The dream to create machines that are skilled and intelligent has been part of humanity from the beginning of time. This dream is now becoming part of our world's striking reality.

Since the early civilizations, one of man's greatest ambitions has been to create artifacts in their image. The legend of the Titan Prometheus, who molded humankind from clay, or that of the giant Talus, the bronze slave forged by Hephaestus (3500 BC), testify to this quest in Greek mythology. The Egyptians' oracle statues hiding priests inside (2500 BC) were perhaps the precursor of our modern thinking machines. The clepsydra water clock introduced by the Babylonians (1400 BC) was one of the first automated mechanical artifacts. In the following centuries, human creativity has given rise to a host of devices such as the automaton theatre of Hero of Alexandria (100 AD), the hydro-powered water-raising and humanoid machines of Al-Jazari (1200), and Leonardo da Vinci's numerous ingenious designs (1500). The development of automata continued to flourish in the eighteenth century both in Europe and Asia, with creations such as Jacquet-Droz's family of androids (drawer, musician and writer) and the "karakuri-ningyo" mechanical dolls (tea server and archer).

The robot "concept" was clearly established by those many creative historical realizations. Nonetheless, the emergence of the "physical" robot had to await the advent of its underlying technologies during the course of the twentieth century. In 1920, the term *robot* – derived from "robota" which means subordinate labour in Slav languages – was first introduced by the Czech playwright Karel Čapek in his play "Rossum's Universal Robots (R.U.R.)". In 1940, the ethics of the interaction

between robots and humans was envisioned to be governed by the well-known three fundamental laws of Isaac Asimov, the Russian science-fiction writer in his novel "Runaround".

The middle of the twentieth century brought the first explorations of the connection between human intelligence and machines, marking the beginning of an era of fertile research in the field of artificial intelligence (AI). Around that time, the first robots were realized. They benefited from advances in the different technologies of mechanics, controls, computers and electronics. As always, new designs motivate new research and discoveries, which, in turn, lead to enhanced solutions and thus to novel concepts. This virtuous circle over time produced that knowledge and understanding which gave birth to the field of *robotics*, properly referred to as: the science and technology of robots.

The early robots built in the 1960s stemmed from the confluence of two technologies: numerical control machines for precise manufacturing, and teleoperators for remote radioactive material handling. These master-slave arms were designed to duplicate one-to-one the mechanics of the human arm, and had rudimentary control and little perception about the environment. Then, during the mid-to-late twentieth century, the development of integrated circuits, digital computers and miniaturized components enabled computer-controlled robots to be designed and programmed. These robots, termed industrial robots, became essential components in the automation of flexible manufacturing systems in the late 1970s. Further to their wide application in the automotive industry, industrial robots were successfully employed in general industry, such as the metal products, the chemical, the electronics and the food industries. More recently, robots have found new applications outside the factories, in areas such as cleaning, search and rescue, underwater, space, and medical applications.

In the 1980s robotics was defined as the science which studies the *intelligent connection between perception and action*. With reference to this definition, the action of a robotic system is entrusted to a *locomotion*

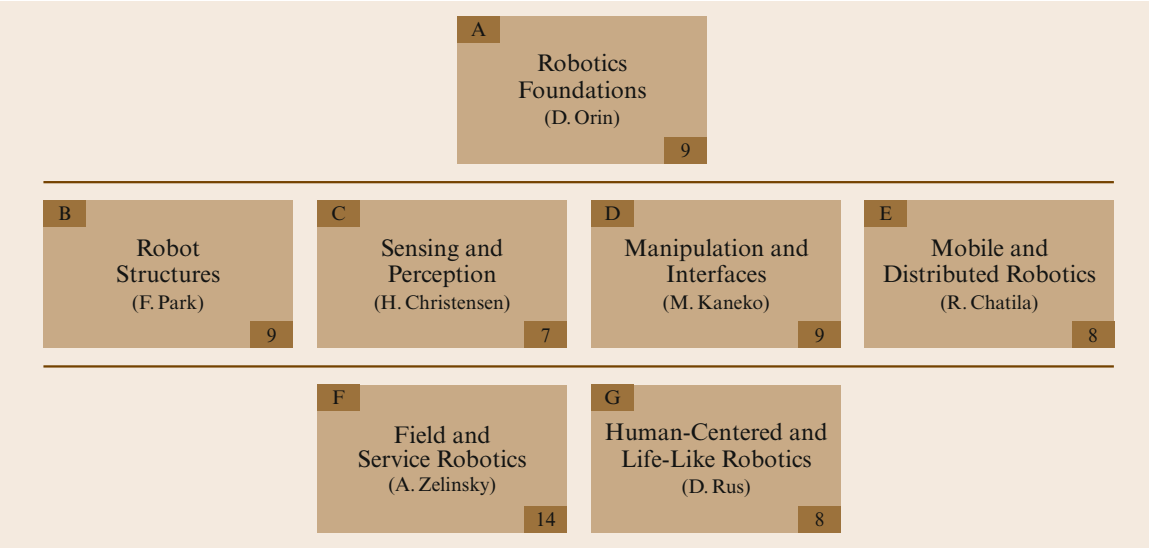


Fig. 1 Organization of the handbook

apparatus to move in the environment (wheels, crawlers, legs, propellers) and/or to a *manipulation* apparatus to operate on objects present in the environment (arms, end effectors, artificial hands), where suitable *actuators* animate the mechanical components of the robot. The perception is extracted from the *sensors* providing information on state of the robot (position and speed) and its surrounding environment (force and tactile, range and vision). The intelligent connection is entrusted to a *programming, planning and control* architecture which relies on the perception and available *models* of the robot and environment and exploits learning and skill acquisition.

In the 1990s research was boosted by the need to resort to robots to address human safety in hazardous environments (*field robotics*), or to enhance the human operator ability and reduce his/her fatigue (*human augmentation*), or else by the desire to develop products with wide potential markets aimed at improving the quality of life (*service robotics*). A common denominator of such application scenarios was the need to operate in a scarcely structured environment which ultimately requires increased abilities and a higher degree of *autonomy*.

By the dawn of the new millennium, robotics has undergone a major transformation in scope and dimensions. This expansion has been brought about by the maturity of the field and the advances in its related technologies. From a largely dominant industrial focus, robotics has been rapidly expanding into the

challenges of the human world (*human-centered and life-like robotics*). The new generation of robots is expected to safely and dependably co-habitat with humans in homes, workplaces, and communities, providing support in services, entertainment, education, healthcare, manufacturing, and assistance.

Beyond its impact on physical robots, the body of knowledge robotics has produced is revealing a much wider range of applications reaching across diverse research areas and scientific disciplines, such as: biomechanics, haptics, neurosciences, virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. It is indeed at the intersection of disciplines that the most striking advances happen.

Today, new communities of users and developers are forming, with growing connections to the core of robotics research. A strategic goal for the robotics community is one of outreach and scientific cooperation with these communities. Future developments and expected growth of the field will largely depend on the research community's abilities to achieve this objective.

The dissemination of research results and findings in archival publications and conference presentations has played an important role in the advancement of robotics in the past decades. The extent of scientific activities in robotics has led to the establishment of professional societies and research networks devoted to the field.

The introduction of graduate programs in robotics in many academic institutions around the world is a clear illustration of the level of maturity reached by robotics as a scientific field.

The intensive stream of robotics research documented in the literature culminates into this unique reference, which aims at collecting in one self-contained volume the most significant achievements of our international robotics community. The *Springer Handbook of Robotics* presents a full coverage of the field from its foundations, through the research areas, up to the new emerging applications of robotics. Accordingly, the material is organized in three logical layers reflecting the historical development of the field, as illustrated in Fig. 1.

The foundations of robotics science, laid down in the first layer (Part A with its 9 chapters), address the theory of robot mechanics, sensing, planning, and control. The consolidated methodologies and technologies of robot structures (Part B with its 9 chapters), sensing and perception (Part C with its 7 chapters), manipulation and interfaces (Part D with its 8 chapters) and mobile and distributed robotics (Part E with its 8 chapters) are presented in the second layer. The third layer is devoted to advanced applications, such as in field and service robotics (Part F with its 14 chapters) and human-centered and life-like robotics (Part G with its 9 chapters).

Part A presents the fundamental principles and methods that are used to model, design, and control a robotic system. All of the foundational topics are included in this part: kinematics, dynamics, mechanical design and actuation, sensing and estimation, motion planning, motion control, force control, robotic systems architectures and programming, and **AI** reasoning methods for task planning and learning. A chapter is devoted to each of these topics. The topics are expanded and applied to specific robotic structures and systems in subsequent parts.

Part B is concerned with the design, modeling, motion planning, and control of the actual physical realizations of a robot. Some of the more obvious mechanical structures that come to mind are arms, legs, and hands; to this list can be added wheeled vehicles and platforms, and robot structures at the micro and nano scales. With separate chapters devoted to performance criteria and model identification, the chapters in this part successively examine serial redundant mechanisms, parallel

mechanisms, flexible robots, robot hands, robot legs, wheeled robots, and micro and nano-scale robots.

Part C covers different sensory modalities and integration of sensor data across space and time to generate models of robots and the external environment. Robotics is the intelligent coupling of perception and action and as such Part C complements Part B to build systems. This part of the handbook covers sensing across contact, proprioception to exteroception. The main sensor types such as tactile, odometry, **GPS**, ranging and visual perception are presented. Both basic sensor models, sensor data processing and associated representations are covered. Finally, a chapter on sensor fusion introduces the mathematical tools needed for integration of sensor information across space and time.

Part D is concerned with interaction between robots and objects, or between humans and robots. Manipulation is supposed to handle an object through direct contact by arms or fingers or just a pusher, while interfaces are supposed to make either direct or indirect interaction between humans and robots. For enhancing dexterity in robot manipulation, motion for manipulation tasks, contact modeling and manipulation, grasping, and cooperative manipulators are addressed in the first half of this part. For achieving a skillful manipulation or power increase in a human/robot system, haptics, telerobotics, networked telerobotics, and exoskeletons for human performance augmentation are discussed in the second half of part D.

Part E covers a wide span of topics. This part addresses motion planning and control of wheeled robots with kinematic constraints, perception and world modeling, simultaneous localization and mapping, and the integration of those capacities in a control architecture, as a mobile robot is actually the paradigm of a complex integrated system. This part completes Part A on foundations in the context of mobile robotics, and given the role of perception, is closely related to Part C on sensing. In addition, multi-robot interaction and systems are also discussed, including modular and reconfigurable robots as well as networked robotics.

Part F covers topics related to creating field and service application-based robots that operate in all types of environments. This includes applications ranging from industrial robots, through a diverse array of air, land, sea and space applications to educational robotics. This part

of the handbook, which draws on Parts A, B, C, D and E, describes how robots can be put to work.

Part G covers topics related to creating robots that operate in human-centered environments. This includes the design of robots with humanoid and other biologically inspired morphologies, sensors, actuators, and control architectures. User interfaces, such as programming by demonstration and programming for safety, are also included in this part, and it concludes with the socio-ethical implications of robots.

The handbook was conceived to provide a valuable resource not only for robotics experts, but also for newcomers to this expanding field (engineers, medical doctors, computer scientists, and designers). In particular, it is important to underline the tutorial value of Part A to graduate students and post-docs, the research value of Parts B to E for a wider coverage of research in robotics, and the added value of Parts F and G to engineers and scientists interested in new applications.

The contents of the various chapters have been inspired by a classic cut, i.e. avoiding the inclusion of on-going or not well-established methods. An objective perspective has been taken, while covering multiple approaches, with the goal of ensuring a high archival value to the handbook. Each chapter is preceded by a short summary, and an introductory section providing the state of the art in the area. The core sections are developed at a tutorial level. Lengthy mathematical derivations have been avoided whenever possible, while the equations, tables, and algorithms are illustrated in ready-to-use form. The final section of the chapter provides conclusions and topics for further reading.

From the foundations to the social and ethical implications of robotics, the sixty-four chapters of the handbook provide a comprehensive collection of five decades of progress in robotics. This sum is a testimony to the level of accomplishments in our field, and a premise of further advances towards new frontiers of robotics.

Robotics

Part A

Part A Robotics Foundations

Ed. by David E. Orin

1 Kinematics

Kenneth Waldron, Stanford, USA
James Schmiedeler, Columbus, USA

2 Dynamics

Roy Featherstone, Canberra, Australia
David E. Orin, Columbus, USA

3 Mechanisms and Actuation

Victor Scheinman, Stanford, USA
J. Michael McCarthy, Irvine, USA

4 Sensing and Estimation

Henrik I. Christensen, Atlanta, USA
Gregory D. Hager, Baltimore, USA

5 Motion Planning

Lydia E. Kavradi, Houston, USA
Steven M. LaValle, Urbana, USA

6 Motion Control

Wankyun Chung, Pohang, Korea
Li-Chen Fu, Taipei, Taiwan, R.O.C.
Su-Hau Hsu[†]

7 Force Control

Luigi Villani, Napoli, Italy
Joris De Schutter, Leuven-Heverlee, Belgium

8 Robotic Systems Architectures and Programming

David Kortenkamp, Houston, USA
Reid Simmons, Pittsburgh, USA

9 AI Reasoning Methods for Robotics

Joachim Hertzberg, Osnabrück, Germany
Raja Chatila, Toulouse, France

The chapters contained in **Part A**, Robotics Foundations, present the fundamental principles and methods that are used to develop a robotic system. In order to perform the tasks that are envisioned for robots, many challenging problems have been uncovered in kinematics, dynamics, design, actuation, sensing, motion planning, control, programming, and task planning. The chapters in this part address the basic issues in each of these areas. Some of the basic problems in robotics are outlined as follows. Robots often consist of a large number of degrees of freedom so that they can provide the rich set of three-dimensional (3-D) motions that may be required for a range of tasks. The kinematic and dynamic relationships between the joint actuators' motion and torques, and the desired motion and force for a task can be very complex. The design of the link and joint structures, as well as the actuation, to achieve the desired performance is also challenging. The robot is a nonlinear, coupled system which is difficult to control because of its complex dynamics. This is exacerbated when the environment is unstructured, and often sophisticated sensing and estimation techniques are required.

In addition to control of the motion, control of the interaction forces between the robot and environment is needed when manipulating objects or interacting with humans. A fundamental robotics task is to plan collision-free motion for complex bodies from a start to a goal position among a collection of obstacles, and this can become an intractable computational problem. In order to achieve some of the intelligence ascribed to humans, robots need to be equipped with sophisticated task planners that employ symbolic reasoning to move in dynamic, partially known environments. Robot software architectures also have special needs because of these requirements.

While the basic issues outlined in the previous paragraphs are addressed in this part, more depth can be found in other parts of the Handbook. The kinematics, dynamics, mechanical design, and control principles and methods introduced in this part can be applied to robotic structures made up of arms, hands, and legs (Part B) as well as to manipulators (Part D), wheeled mobile robots (Part E), and field and service robots (Part F). Force control is especially important for manipulators and their interfaces (Part D). The basic sensing and estimation techniques presented here are expanded and applied to specific sensing modalities in Part C. Motion planning is an important aspect of manipulation (Part D) and mobile and distributed robotic systems (Part E). Robotic systems architectures and AI reasoning methods are particularly important in mobile and

distributed robotics (Part E) and human-centered and lifelike robotics (Part G).

With this overview of Part A, we now provide a brief synopsis of each chapter.

Chapter 1, Kinematics, provides a number of representations and conventions to describe the motion of the bodies in a robotic mechanism. These include rotation matrices, Euler angles, quaternions, homogeneous transformations, screw transformations, matrix exponential parameterization, and Plücker coordinates. Representations of the kinematics of all common joint types are provided, along with a modified form of the Denavit–Hartenberg convention. These representational tools are applied to compute the workspace, the forward and inverse kinematics, the forward and inverse instantaneous kinematics, the Jacobian, and the static wrench transmission.

Chapter 2, Dynamics, presents the dynamic equations of motion which provide the relationships between actuation and contact forces acting on robot mechanisms, and the acceleration and motion trajectories that result. Efficient algorithms are provided for important dynamics computations which include inverse dynamics, forward dynamics, the joint-space inertia matrix, and the operational-space inertia matrix. The algorithms may be applied to fixed-base robots, mobile robots, and parallel robot mechanisms. Compact formulation of the algorithms results from using six-dimensional (6-D) spatial notation to describe rigid-body velocity, acceleration, inertia, etc.

Chapter 3, Mechanisms and Actuation, focuses on the principles that guide the design and construction of robotic systems. The kinematic equations and Jacobian are used to characterize the work envelope and mechanical advantage, and guide the selection of the robot's size and joint arrangement. The design of both serial and parallel robots is addressed. Practical consideration is given to the design of the link and joint structures along with selection of the actuators and transmission drives to power the movement. Robot performance in terms of speed, acceleration, repeatability, and other measures is also addressed.

Chapter 4, Sensing and Estimation, provides a brief overview of common sensing methods and estimation techniques that have found broad applicability in robotics. These provide information about the state of the environment and robot system. The presentation is structured according to a perception process model that includes sensing, feature extraction, data association, parameter estimation, and model integration. Several common sensing modalities are introduced and charac-

terized. Methods for estimation in linear and nonlinear systems are discussed, including statistical estimation, the Kalman filter, and sample-based methods. The several common representations for estimation are also introduced.

Chapter 5, Motion Planning, addresses the fundamental robotics task to plan collision-free motion for complex bodies from a start to a goal position among a collection of obstacles. The basic geometric path planning problem (the piano mover's problem) is introduced, but the focus of the chapter is on sampling-based planning methods because of their generally wider applicability. Planning with differential constraints is considered and is important for wheeled mobile robots. Extensions and variations to the basic motion planning problem, as well as advanced issues, are discussed at the end of the chapter.

Chapter 6, Motion Control, focuses on the motion control of a rigid manipulator. The main challenges addressed are the complexity of the nonlinear, coupled dynamics and the structured and unstructured uncertainties. The chapter discusses topics ranging from independent-joint control and **PID** control to computed-torque control to manage the complex dynamics. Adaptive and robust control are presented to address the problems related to the uncertainties in the system. The chapter concludes with some practical considerations of digital implementation and learning control for repetitive motions.

Chapter 7, Force Control, focuses on the control of the interaction forces between a robotic system and its

environment. The chapter groups interaction control into two categories: indirect and direct force control, which achieve the control without (indirect) or with (direct) explicit closure of a force feedback loop. Impedance control and hybrid force/motion control are examples of each, respectively. The fundamental problem of interaction tasks modeling is presented to provide the basis for the force control schemes.

Chapter 8, Robotic Systems Architectures, and Programming, presents the software architectures and supporting programming tools and environments that have been developed for robotic systems. Robot architectures have special needs because of the requirements of the robot to interact asynchronously, in real time, with an uncertain, often dynamic, environment. The chapter discusses the major types of architectural components for a layered robot control architecture – behavioral control, executives, and task planners – along with the commonly used techniques for interconnecting these components.

Chapter 9, AI Reasoning Methods for Robotics, sketches the state of the art in artificial intelligence for the symbol-based reasoning methods and applications considered most relevant for robotics. Reasoning on a mobile robot is especially challenging because of its dynamic, partially known environment. The chapter describes knowledge representation and inference, covering both logics- and probability-based approaches. Besides reasoning methods, the chapter considers generic reasoning applications, namely, action planning and robot learning.

Kinematics

1. Kinematics

Kenneth Waldron, James Schmiedeler

Kinematics pertains to the motion of bodies in a robotic mechanism without regard to the forces/torques that cause the motion. Since robotic mechanisms are by their very essence designed for motion, kinematics is the most fundamental aspect of robot design, analysis, control, and simulation. The robotics community has focused on efficiently applying different representations of position and orientation and their derivatives with respect to time to solve foundational kinematics problems.

This chapter will present the most useful representations of the position and orientation of a body in space, the kinematics of the joints most commonly found in robotic mechanisms, and a convenient convention for representing the geometry of robotic mechanisms. These representational tools will be applied to compute the *workspace*, the *forward* and *inverse kinematics*, the *forward* and *inverse instantaneous kinematics*, and the *static wrench transmission* of a robotic mechanism. For brevity, the focus will be on algorithms applicable to open-chain mechanisms.

The goal of this chapter is to provide the reader with general tools in tabulated form and a broader overview of algorithms that can be applied together to solve kinematics problems pertaining to a particular robotic mechanism.

1.1	Overview	9
1.2	Position and Orientation Representation	10
1.2.1	Position and Displacement.....	10
1.2.2	Orientation and Rotation	10
1.2.3	Homogeneous Transformations.....	13
1.2.4	Screw Transformations.....	14
1.2.5	Matrix Exponential Parameterization	16
1.2.6	Plücker Coordinates.....	18
1.3	Joint Kinematics	18
1.3.1	Lower Pair Joints.....	19
1.3.2	Higher Pair Joints.....	21
1.3.3	Compound Joints	22
1.3.4	6-DOF Joint	22
1.3.5	Physical Realization.....	22
1.3.6	Holonomic and Nonholonomic Constraints	23
1.3.7	Generalized Coordinates	23
1.4	Geometric Representation	23
1.5	Workspace	25
1.6	Forward Kinematics	26
1.7	Inverse Kinematics	27
1.7.1	Closed-Form Solutions.....	27
1.7.2	Numerical Methods	28
1.8	Forward Instantaneous Kinematics	29
1.8.1	Jacobian	29
1.9	Inverse Instantaneous Kinematics	30
1.9.1	Inverse Jacobian	30
1.10	Static Wrench Transmission	30
1.11	Conclusions and Further Reading	31
	References	31

1.1 Overview

Unless explicitly stated otherwise, robotic mechanisms are systems of rigid bodies connected by joints. The position and orientation of a rigid body in space are collectively termed the *pose*. Therefore, robot kinemat-

ics describes the pose, velocity, acceleration, and all higher-order derivatives of the pose of the bodies that comprise a mechanism. Since kinematics does not address the forces/torques that induce motion, this chapter

focuses on describing pose and velocity. These descriptions are foundational elements of dynamics (Chap. 2), motion planning (Chap. 5), and motion control (Chap. 6) algorithms.

Among the many possible topologies in which systems of bodies can be connected, two are of particular importance in robotics: serial chains and fully parallel mechanisms. A serial chain is a system of rigid bodies in

which each member is connected to two others, except for the first and last members that are each connected to only one other member. A fully parallel mechanism is one in which there are two members that are connected together by multiple joints. In practice, each *joint* is often itself a serial chain. This chapter focuses almost exclusively on algorithms applicable to serial chains. Parallel mechanisms are dealt with in more detail in Chap. 12.

1.2 Position and Orientation Representation

Spatial, rigid-body kinematics can be viewed as a comparative study of different ways of representing the pose of a body. Translations and rotations, referred to in combination as rigid-body displacements, are also expressed with these representations. No one approach is optimal for all purposes, but the advantages of each can be leveraged appropriately to facilitate the solution of different problems.

The minimum number of coordinates required to locate a body in Euclidean space is six. Many representations of spatial pose employ sets with superabundant coordinates in which auxiliary relationships exist among the coordinates. The number of independent auxiliary relationships is the difference between the number of coordinates in the set and six.

This chapter and those that follow it make frequent use of *coordinate reference frames* or simply *frames*. A coordinate reference frame i consists of an origin, denoted O_i , and a triad of mutually orthogonal basis vectors, denoted $(\hat{x}_i \hat{y}_i \hat{z}_i)$, that are all fixed within a particular body. The pose of a body will always be expressed relative to some other body, so it can be expressed as the pose of one coordinate frame relative to another. Similarly, rigid-body displacements can be expressed as displacements between two coordinate frames, one of which may be referred to as *moving*, while the other may be referred to as *fixed*. This indicates that the observer is located in a stationary position within the fixed reference frame, not that there exists any absolutely fixed frame.

1.2.1 Position and Displacement

The position of the origin of coordinate frame i relative to coordinate frame j can be denoted by the 3×1 vector

$${}^j\mathbf{p}_i = \begin{pmatrix} {}^j p_i^x \\ {}^j p_i^y \\ {}^j p_i^z \end{pmatrix}.$$

The components of this vector are the Cartesian coordinates of O_i in the j frame, which are the projections of the vector ${}^j\mathbf{p}_i$ onto the corresponding axes. The vector components could also be expressed as the spherical or cylindrical coordinates of O_i in the j frame. Such representations have advantages for analysis of robotic mechanisms including spherical and cylindrical joints.

A translation is a displacement in which no point in the rigid body remains in its initial position and all straight lines in the rigid body remain parallel to their initial orientations. (The points and lines are not necessarily contained within the boundaries of the finite rigid body, but rather, any point or line in space can be taken to be rigidly fixed in a body.) The translation of a body in space can be represented by the combination of its positions prior to and following the translation. Conversely, the position of a body can be represented as a translation that takes the body from a position in which the coordinate frame fixed to the body coincides with the fixed coordinate frame to the current position in which the two frames are not coincident. Thus, any representation of position can be used to create a representation of displacement, and vice versa.

1.2.2 Orientation and Rotation

There is significantly greater breadth in the representation of orientation than in that of position. This section does not include an exhaustive summary, but focuses on the representations most commonly applied to robotic mechanisms.

A rotation is a displacement in which at least one point of the rigid body remains in its initial position and not all lines in the body remain parallel to their initial orientations. For example, a body in a circular orbit rotates about an axis through the center of its circular path, and every point on the axis of rotation is a point

in the body that remains in its initial position. As in the case of position and translation, any representation of orientation can be used to create a representation of rotation, and vice versa.

Rotation Matrices

The orientation of coordinate frame i relative to coordinate frame j can be denoted by expressing the basis vectors ($\hat{x}_i \hat{y}_i \hat{z}_i$) in terms of the basis vectors ($\hat{x}_j \hat{y}_j \hat{z}_j$). This yields (${}^j\hat{x}_i {}^j\hat{y}_i {}^j\hat{z}_i$), which when written together as a 3×3 matrix is known as the rotation matrix. The components of ${}^j\mathbf{R}_i$ are the dot products of basis vectors of the two coordinate frames.

$${}^j\mathbf{R}_i = \begin{pmatrix} \hat{x}_i \cdot \hat{x}_j & \hat{y}_i \cdot \hat{x}_j & \hat{z}_i \cdot \hat{x}_j \\ \hat{x}_i \cdot \hat{y}_j & \hat{y}_i \cdot \hat{y}_j & \hat{z}_i \cdot \hat{y}_j \\ \hat{x}_i \cdot \hat{z}_j & \hat{y}_i \cdot \hat{z}_j & \hat{z}_i \cdot \hat{z}_j \end{pmatrix}. \quad (1.1)$$

Because the basis vectors are unit vectors and the dot product of any two unit vectors is the cosine of the angle between them, the components are commonly referred to as direction cosines.

An elementary rotation of frame i about the \hat{z}_j axis through an angle θ is

$$\mathbf{R}_Z(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad (1.2)$$

while the same rotation about the \hat{y}_j axis is

$$\mathbf{R}_Y(\theta) = \begin{pmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{pmatrix}, \quad (1.3)$$

and about the \hat{x}_j axis is

$$\mathbf{R}_X(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{pmatrix}. \quad (1.4)$$

The rotation matrix ${}^j\mathbf{R}_i$ contains nine elements, while only three parameters are required to define the orientation of a body in space. Therefore, six auxiliary relationships exist between the elements of the matrix. Because the basis vectors of coordinate frame i are mutually orthonormal, as are the basis vectors of coordinate frame j , the columns of ${}^j\mathbf{R}_i$ formed from the dot products of these vectors are also mutually orthonormal. A matrix composed of mutually orthonormal vectors is known as an orthogonal matrix and has the property that its inverse is simply its transpose. This property provides the six auxiliary relationships. Three require the

Table 1.1 Equivalent rotation matrices for various representations of orientation, with abbreviations $c_\theta := \cos \theta$, $s_\theta := \sin \theta$, and $v_\theta := 1 - \cos \theta$

Z-Y-X Euler angles (α, β, γ):

$${}^j\mathbf{R}_i = \begin{pmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta c_\gamma + s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta c_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{pmatrix}$$

X-Y-Z fixed angles (ψ, θ, ϕ):

$${}^j\mathbf{R}_i = \begin{pmatrix} c_\phi c_\theta & c_\phi s_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ s_\phi c_\theta & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{pmatrix}$$

Angle-axis $\theta \hat{\mathbf{w}}$:

$${}^j\mathbf{R}_i = \begin{pmatrix} w_x^2 v_\theta + c_\theta & w_x w_y v_\theta - w_z s_\theta & w_x w_z v_\theta + w_y s_\theta \\ w_x w_y v_\theta + w_z s_\theta & w_y^2 v_\theta + c_\theta & w_y w_z v_\theta - w_x s_\theta \\ w_x w_z v_\theta - w_y s_\theta & w_y w_z v_\theta + w_x s_\theta & w_z^2 v_\theta + c_\theta \end{pmatrix}$$

Unit quaternions ($\epsilon_0 \epsilon_1 \epsilon_2 \epsilon_3$)^T:

$${}^j\mathbf{R}_i = \begin{pmatrix} 1 - 2(\epsilon_2^2 + \epsilon_3^2) & 2(\epsilon_1 \epsilon_2 - \epsilon_0 \epsilon_3) & 2(\epsilon_1 \epsilon_3 + \epsilon_0 \epsilon_2) \\ 2(\epsilon_1 \epsilon_2 + \epsilon_0 \epsilon_3) & 1 - 2(\epsilon_1^2 + \epsilon_3^2) & 2(\epsilon_2 \epsilon_3 - \epsilon_0 \epsilon_1) \\ 2(\epsilon_1 \epsilon_3 - \epsilon_0 \epsilon_2) & 2(\epsilon_2 \epsilon_3 + \epsilon_0 \epsilon_1) & 1 - 2(\epsilon_1^2 + \epsilon_2^2) \end{pmatrix}$$

column vectors to have unit length, and three require the column vectors to be mutually orthogonal. Alternatively, the orthogonality of the rotation matrix can be seen by considering the frames in reverse order. The orientation of coordinate frame j relative to coordinate frame i is the rotation matrix ${}^i\mathbf{R}_j$ whose rows are clearly the columns of the matrix ${}^j\mathbf{R}_i$. Rotation matrices are combined through simple matrix multiplication such that the orientation of frame i relative to frame k can be expressed as

$${}^k\mathbf{R}_i = {}^k\mathbf{R}_j {}^j\mathbf{R}_i.$$

In summary, ${}^j\mathbf{R}_i$ is the rotation matrix that transforms a vector expressed in coordinate frame i to a vector expressed in coordinate frame j . It provides a representation of the orientation of frame i relative to j and thus, can be a representation of rotation from frame i to frame j . Table 1.1 lists the equivalent rotation matrices for the other representations of orientation listed in this section. Table 1.2 contains the conversions from a known rotation matrix to these other representations.

Euler Angles

For a minimal representation, the orientation of coordinate frame i relative to coordinate frame j can be denoted as a vector of three angles (α, β, γ). These angles are known as Euler angles when each represents a rotation about an axis of a moving coordinate frame. In this way, the location of the axis of each successive rotation depends upon the preceding rotation(s), so the order of the

Table 1.2 Conversions from a rotation matrix to various representations of orientation

Rotation matrix:
${}^jR_i = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$
Z-Y-X Euler angles (α, β, γ) :
$\beta = \text{Atan2}(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2})$
$\alpha = \text{Atan2}\left(\frac{r_{21}}{\cos \beta}, \frac{r_{11}}{\cos \beta}\right)$
$\gamma = \text{Atan2}\left(\frac{r_{32}}{\cos \beta}, \frac{r_{33}}{\cos \beta}\right)$
X-Y-Z fixed angles (ψ, θ, ϕ) :
$\theta = \text{Atan2}(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2})$
$\psi = \text{Atan2}\left(\frac{r_{21}}{\cos \theta}, \frac{r_{11}}{\cos \theta}\right)$
$\phi = \text{Atan2}\left(\frac{r_{32}}{\cos \theta}, \frac{r_{33}}{\cos \theta}\right)$
Angle axis $\theta \hat{\mathbf{w}}$:
$\theta = \cos^{-1} \left(\frac{r_{11} + r_{22} + r_{33} - 1}{2} \right)$
$\hat{\mathbf{w}} = \frac{1}{2 \sin \theta} \begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix}$
Unit quaternions $(\epsilon_0 \ \epsilon_1 \ \epsilon_2 \ \epsilon_3)^\top$:
$\epsilon_0 = \frac{1}{2} \sqrt{1 + r_{11} + r_{22} + r_{33}}$
$\epsilon_1 = \frac{r_{32} - r_{23}}{4\epsilon_0}$
$\epsilon_2 = \frac{r_{13} - r_{31}}{4\epsilon_0}$
$\epsilon_3 = \frac{r_{21} - r_{12}}{4\epsilon_0}$

rotations must accompany the three angles to define the orientation. For example, the symbols (α, β, γ) are used throughout this handbook to indicate Z-Y-X Euler angles. Taking the moving frame i and the fixed frame j to be initially coincident, α is the rotation about the $\hat{\mathbf{z}}$ axis of frame i , β is the rotation about the rotated $\hat{\mathbf{y}}$ axis of frame i , and finally, γ is the rotation about the twice rotated $\hat{\mathbf{x}}$ axis of frame i . The equivalent rotation matrix jR_i is given in Table 1.1. Z-Y-Z and Z-X-Z Euler angles are other commonly used conventions from among the 12 different possible orders of rotations.

Regardless of the order of rotations, an Euler angle representation of orientation always exhibits a singularity when the first and last rotations both occur about the same axis. This can be readily seen in Table 1.2 wherein the angles α and γ are undefined when $\beta = \pm 90^\circ$. (For Z-Y-Z and Z-X-Z Euler angles, the singularity occurs when the second rotation is 0° or 180° .) This creates a problem in relating the angular velocity vector of a body to the time derivatives of Euler angles, which somewhat lim-

its their usefulness in modeling robotic systems. This velocity relationship for Z-Y-X Euler angles is

$$\begin{pmatrix} \dot{\alpha} \\ \dot{\beta} \\ \dot{\gamma} \end{pmatrix} = \begin{pmatrix} -\sin \beta & 0 & 1 \\ \cos \beta \sin \gamma & \cos \gamma & 0 \\ \cos \beta \cos \gamma & -\sin \beta & 0 \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}. \quad (1.5)$$

Fixed Angles

A vector of three angles can also denote the orientation of coordinate frame i relative to coordinate frame j when each angle represents a rotation about an axis of a fixed reference frame. Appropriately, such angles are referred to as fixed angles, and the order of the rotations must again accompany the angles to define the orientation. X-Y-Z fixed angles, denoted here as (ψ, θ, ϕ) , are a common convention from among the, again, 12 different possible orders of rotations. Taking the moving frame i and the fixed frame j to be initially coincident, ψ is the yaw rotation about the fixed $\hat{\mathbf{x}}_j$ axis, θ is the pitch rotation about the fixed $\hat{\mathbf{y}}_j$ axis, and ϕ is the roll rotation about the fixed $\hat{\mathbf{z}}_j$ axis. As can be seen by comparing the respective equivalent rotation matrices in Table 1.1 and the respective conversions in Table 1.2, a set of X-Y-Z fixed angles is exactly equivalent to the same set of Z-Y-X Euler angles ($\alpha = \phi$, $\beta = \theta$, and $\gamma = \psi$). This result holds in general such that three rotations about the three axes of a fixed frame define the same orientation as the same three rotations taken in the opposite order about the three axes of a moving frame. Likewise, all fixed-angle representations of orientations suffer from the singularity discussed for Euler angles. Also, the relationship between the time derivatives of fixed angles and the angular velocity vector is similar to the relationship for Euler angles.

Angle-Axis

A single angle θ in combination with a unit vector $\hat{\mathbf{w}}$ can also denote the orientation of coordinate frame i relative to coordinate frame j . In this case, frame i is rotated through the angle θ about an axis defined by the vector $\hat{\mathbf{w}} = (w_x \ w_y \ w_z)^\top$ relative to frame j . The vector $\hat{\mathbf{w}}$ is sometimes referred to as the equivalent axis of a finite rotation. The angle-axis representation, typically written as either $\theta \hat{\mathbf{w}}$ or $(\theta w_x \ \theta w_y \ \theta w_z)^\top$, is superabundant by one because it contains four parameters. The auxiliary relationship that resolves this is the unit magnitude of vector $\hat{\mathbf{w}}$. Even with this auxiliary relationship, the angle-axis representation is not unique because rotation through an angle of $-\theta$ about $-\hat{\mathbf{w}}$ is equivalent to a rotation through θ about $\hat{\mathbf{w}}$. Table 1.3 contains

Table 1.3 Conversions from angle-axis to unit quaternion representations of orientation and vice versaAngle-axis $\theta \hat{\mathbf{w}}$ to unit quaternion $(\epsilon_0 \ \epsilon_1 \ \epsilon_2 \ \epsilon_3)^\top$:

$$\begin{aligned}\epsilon_0 &= \cos \frac{\theta}{2} \\ \epsilon_1 &= w_x \sin \frac{\theta}{2} \\ \epsilon_2 &= w_y \sin \frac{\theta}{2} \\ \epsilon_3 &= w_z \sin \frac{\theta}{2}\end{aligned}$$

Unit quaternion $(\epsilon_0 \ \epsilon_1 \ \epsilon_2 \ \epsilon_3)^\top$ to angle-axis $\theta \hat{\mathbf{w}}$:

$$\begin{aligned}\theta &= 2 \cos^{-1} \epsilon_0 \\ w_x &= \frac{\epsilon_1}{\sin \frac{\theta}{2}} \\ w_y &= \frac{\epsilon_2}{\sin \frac{\theta}{2}} \\ w_z &= \frac{\epsilon_3}{\sin \frac{\theta}{2}}\end{aligned}$$

the conversions from angle-axis representation to unit quaternions and vice versa. The conversions from these two representations to Euler angles or fixed angles can be easily found by using the conversions in Table 1.2 in conjunction with the equivalent rotation matrices in Table 1.1. Velocity relationships are more easily dealt with using the closely related quaternion representation.

Quaternions

The quaternion representation of orientation due to *Hamilton* [1.1], while largely superseded by the simpler vector representations of *Gibbs* [1.2] and *Grassmann* [1.3], is extremely useful for problems in robotics that result in representational singularities in the vector/matrix notation [1.4]. Quaternions do not suffer from singularities as Euler angles do.

A quaternion ϵ is defined to have the form

$$\epsilon = \epsilon_0 + \epsilon_1 i + \epsilon_2 j + \epsilon_3 k,$$

where the components ϵ_0 , ϵ_1 , ϵ_2 , and ϵ_3 are scalars, sometimes referred to as Euler parameters, and i , j , and k are operators. The operators are defined to satisfy the following combinatory rules:

$$\begin{aligned}ii &= jj = kk = -1, \\ ij &= k, \quad jk = i, \quad ki = j, \\ ji &= -k, \quad kj = -i, \quad ik = -j.\end{aligned}$$

Two quaternions are added by adding the respective components separately, so the operators act as separators. The null element for addition is the quaternion $\mathbf{0} = 0 + 0i + 0j + 0k$, and quaternion sums are associative, commutative, and distributive. The null element for multiplication is $\mathbf{1} = 1 + 0i + 0j + 0k$, as can be seen

using $\mathbf{1}\epsilon = \epsilon$ for any quaternion ϵ . Quaternion products are associative and distributive, but not commutative, and following the conventions of the operators and addition, have the form

$$\begin{aligned}\mathbf{ab} &= a_0 b_0 - a_1 b_1 - a_2 b_2 - a_3 b_3 \\ &\quad + (a_0 b_1 + a_1 b_0 + a_2 b_3 - a_3 b_2)i \\ &\quad + (a_0 b_2 + a_2 b_0 + a_3 b_1 - a_1 b_3)j \\ &\quad + (a_0 b_3 + a_3 b_0 + a_1 b_2 - a_2 b_1)k.\end{aligned}\quad (1.6)$$

It is convenient to define the conjugate of a quaternion

$$\tilde{\epsilon} = \epsilon_0 - \epsilon_1 i - \epsilon_2 j - \epsilon_3 k,$$

so that

$$\epsilon \tilde{\epsilon} = \tilde{\epsilon} \epsilon = \epsilon_0^2 + \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2.$$

A unit quaternion can then be defined such that $\epsilon \tilde{\epsilon} = 1$. Often, ϵ_0 is referred to as the scalar part of the quaternion, and $(\epsilon_1 \ \epsilon_2 \ \epsilon_3)^\top$ is referred to as the vector part.

Unit quaternions are used to describe orientation, and the unit magnitude provides the auxiliary relationship to resolve the use of superabundant (four) coordinates. A vector is defined in quaternion notation as a quaternion with $\epsilon_0 = 0$. Thus, a vector $\mathbf{p} = (p_x \ p_y \ p_z)^\top$ can be expressed as a quaternion $\mathbf{p} = p_x i + p_y j + p_z k$. For any unit quaternion ϵ , the operation $\epsilon \mathbf{p} \tilde{\epsilon}$ performs a rotation of the vector \mathbf{p} about the direction $(\epsilon_1 \ \epsilon_2 \ \epsilon_3)^\top$. This is clearly seen by expanding the operation $\epsilon \mathbf{p} \tilde{\epsilon}$ and comparing the results with the equivalent rotation matrix listed in Table 1.1. Also, as shown in Table 1.3, unit quaternions are closely related to the angle-axis representation of orientation. ϵ_0 corresponds to the angle of rotation, while ϵ_1 , ϵ_2 , and ϵ_3 define the axis of rotation.

For velocity analysis, the time derivative of the quaternion can be related to the angular velocity vector as

$$\begin{pmatrix} \dot{\epsilon}_0 \\ \dot{\epsilon}_1 \\ \dot{\epsilon}_2 \\ \dot{\epsilon}_3 \end{pmatrix} = \frac{1}{2} \begin{pmatrix} -\epsilon_1 & -\epsilon_2 & -\epsilon_3 \\ \epsilon_0 & \epsilon_3 & -\epsilon_2 \\ -\epsilon_3 & \epsilon_0 & \epsilon_1 \\ \epsilon_2 & -\epsilon_1 & \epsilon_0 \end{pmatrix} \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix}. \quad (1.7)$$

While a unit quaternion represents only the orientation of a body, quaternions may be dualized [1.5–7] to create an algebra that provides a description of the position and orientation of a body in space. Other combined representations are discussed in the following sections.

1.2.3 Homogeneous Transformations

The preceding sections have addressed representations of position and orientation separately. With homogeneous transformations, position vectors and rotation

matrices are combined together in a compact notation. Any vector ${}^i\mathbf{r}$ expressed relative to the i coordinate frame can be expressed relative to the j coordinate frame if the position and orientation of the i frame are known relative to the j frame. Using the notation of Sect. 1.2.1, the position of the origin of coordinate frame i relative to coordinate frame j can be denoted by the vector ${}^j\mathbf{p}_i = ({}^j p_i^x \ {}^j p_i^y \ {}^j p_i^z)^\top$. Using the notation of Sect. 1.2.2, the orientation of frame i relative to frame j can be denoted by the rotation matrix ${}^j\mathbf{R}_i$. Thus,

$${}^j\mathbf{r} = {}^j\mathbf{R}_i {}^i\mathbf{r} + {}^j\mathbf{p}_i. \quad (1.8)$$

This equation can be written

$$\begin{pmatrix} {}^j\mathbf{r} \\ 1 \end{pmatrix} = \begin{pmatrix} {}^j\mathbf{R}_i & {}^j\mathbf{p}_i \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} {}^i\mathbf{r} \\ 1 \end{pmatrix}, \quad (1.9)$$

where

$${}^j\mathbf{T}_i = \begin{pmatrix} {}^j\mathbf{R}_i & {}^j\mathbf{p}_i \\ \mathbf{0}^\top & 1 \end{pmatrix} \quad (1.10)$$

is the 4×4 homogeneous transformation matrix and $({}^j\mathbf{r} \ 1)^\top$ and $({}^i\mathbf{r} \ 1)^\top$ are the homogeneous representations of the position vectors ${}^j\mathbf{r}$ and ${}^i\mathbf{r}$. The matrix ${}^j\mathbf{T}_i$ transforms vectors from coordinate frame i to coordinate frame j . Its inverse ${}^j\mathbf{T}_i^{-1}$ transforms vectors from coordinate frame j to coordinate frame i .

$${}^j\mathbf{T}_i^{-1} = {}^i\mathbf{T}_j = \begin{pmatrix} {}^j\mathbf{R}_i^\top & -{}^j\mathbf{R}_i^\top {}^j\mathbf{p}_i \\ \mathbf{0}^\top & 1 \end{pmatrix}. \quad (1.11)$$

Composition of 4×4 homogeneous transformation matrices is accomplished through simple matrix multiplication, just as in the case of 3×3 rotation matrices. Therefore, ${}^k\mathbf{T}_i = {}^k\mathbf{T}_j {}^j\mathbf{T}_i$. Since matrix multiplications do not commute, the order or sequence is important.

The homogeneous transformation of a simple rotation about an axis is sometimes denoted **Rot** such that a rotation of θ about an axis $\hat{\mathbf{z}}$ is

$$\mathbf{Rot}(\hat{\mathbf{z}}, \theta) = \begin{pmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1.12)$$

Similarly, the homogeneous transformation of a simple translation along an axis is sometimes denoted **Trans** such that a translation of d along an axis $\hat{\mathbf{x}}$ is

$$\mathbf{Trans}(\hat{\mathbf{x}}, d) = \begin{pmatrix} 1 & 0 & 0 & d \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1.13)$$

Homogeneous transformations are particularly attractive when compact notation is desired and/or when ease of programming is the most important consideration. This is not, however, a computationally efficient representation since it introduces a large number of additional multiplications by ones and zeros. Although homogeneous transformation matrices technically contain 16 elements, four are defined to be zero or one, and the remaining elements are composed of a rotation matrix and a position vector. Therefore, the only truly superabundant coordinates come from the rotation matrix component, so the relevant auxiliary relationships are those associated with the rotation matrix.

1.2.4 Screw Transformations

The transformation in (1.8) can be viewed as composed of a rotation between coordinate frames i and j and a separate displacement between those frames. To get from frame i to frame j , one could perform the rotation first, followed by the displacement, or vice versa. Alternatively, the spatial displacement between the frames can be expressed, except in the case of a pure translation, as a rotation about a unique line combined with a translation parallel to that line.

Chasles' Theorem

Chasles' theorem, in the form stated by Chirikjian and Kyatkin [1.8], has two parts. The first states that:

Any displacement of a body in space can be accomplished by means of a translation of a designated point from its initial to its final position, followed by a rotation of the whole body about that point to bring it into its final orientation.

The second part states that:

Any displacement of a body in space can be accomplished by means of a rotation of the body about a unique line in space accompanied by a translation of the body parallel to that line.

Such a line is called a screw axis, and it is this second result that is usually thought of as Chasles' theorem.

The first part of the theorem is almost axiomatic. A designated point in a body anywhere in Euclidean space can be displaced from a given initial position to a given final position. By further requiring that all points in the body traverse the same displacement, the body translates so that the designated point moves from its initial position to its final position. The body can then be rotated about that point into any given final orientation.

The second part of the theorem depends on this representation of a spatial displacement and requires a more complex argument. A preliminary theorem due to Euler allows greater specificity about the rotation of the body: *Any displacement of a body in which one point remains fixed is equivalent to a rotation of that body about a unique axis passing through that point.* Geometrically, embedding three points in the moving body and letting one be the fixed point about which rotation occurs, each of the other two will have initial and final positions. The right bisector planes of the lines joining the initial and final positions in each case necessarily contain the fixed point. Any line in the bisector plane can be the axis of a rotation that carries the corresponding point from its initial to its final position. Therefore, the unique line common to the two bisector planes is such that rotation about it will carry any point in the body from its initial to its final position. The rigidity condition requires that all planes in the body that contain that line rotate through the same angle.

For any rotation of a rigid body described by a rotation matrix jR_i , Euler's theorem states that there is a unique eigenvector $\hat{\mathbf{w}}$ such that

$${}^jR_i \hat{\mathbf{w}} = \hat{\mathbf{w}}, \quad (1.14)$$

where $\hat{\mathbf{w}}$ is a unit vector parallel to the axis of rotation. This expression requires a unit eigenvalue of jR_i corresponding to the eigenvector $\hat{\mathbf{w}}$. The remaining two eigenvalues are $\cos \theta \pm i \sin \theta$, where i is the complex operator and θ is the angle of rotation of the body about the axis.

Combining the first part of Chasles' theorem with Euler's theorem, a general spatial displacement can be expressed as a translation taking a point from its initial to its final position, followed by a unique rotation about a unique axis through that point that carries the body from its initial to its final orientation. Resolving the translation into components in the direction of and orthogonal to the rotation axis, every point in the body has the same displacement component in the direction of the axis because rotation about it does not affect that component. Projected into a plane normal to the rotation axis, the kinematic geometry of the displacement is identical to that of planar motion. Just as there is a unique point in the plane about which a body could be rotated between two given positions, there is a unique point in the projection plane. If the rotation axis is moved to pass through that point, the spatial displacement can be accomplished by a rotation about that axis combined with a translation along it, as stated by the theorem.

The line about which rotation takes place is called the screw axis of the displacement. The ratio of the linear displacement d to the rotation θ is referred to as the pitch h of the screw axis [1.4]. Thus,

$$d = h\theta. \quad (1.15)$$

The screw axis of a pure translation is not unique. Any line parallel to the translation can be regarded as the screw axis, and since the rotation θ is zero, the axis of a translation is said to have infinite pitch.

A screw axis is most conveniently represented in any reference frame by means of a unit vector $\hat{\mathbf{w}}$ parallel to it and the position vector $\boldsymbol{\rho}$ of any point lying on it. Additional specification of the pitch h and the angle of rotation θ completely defines the location of a second coordinate frame relative to the reference frame. Thus, a total of eight coordinates define a screw transformation, which is superabundant by two. The unit magnitude of vector $\hat{\mathbf{w}}$ provides one auxiliary relationship, but in general, there is no second auxiliary relationship because the same screw axis is defined by all points lying on it, which is to say that the vector $\boldsymbol{\rho}$ contains one free coordinate.

Algebraically, a screw displacement is represented by

$${}^j\mathbf{r} = {}^jR_i({}^i\mathbf{r} - \boldsymbol{\rho}) + d\hat{\mathbf{w}} + \boldsymbol{\rho}. \quad (1.16)$$

Comparing this expression to (1.8) yields

$${}^j\mathbf{p}_i = d\hat{\mathbf{w}} + (\mathbf{1}_{3 \times 3} - {}^jR_i)\boldsymbol{\rho}. \quad (1.17)$$

$\mathbf{1}_{3 \times 3}$ denotes the 3×3 identity matrix. An expression for d is easily obtained by taking the inner product of both sides of the equation with $\hat{\mathbf{w}}$.

$$d = \hat{\mathbf{w}}^\top {}^j\mathbf{p}_i. \quad (1.18)$$

The matrix $\mathbf{1}_{3 \times 3} - {}^jR_i$ is singular, so (1.17) cannot be solved to give a unique value of $\boldsymbol{\rho}$, but since $\boldsymbol{\rho}$ can represent any point on the screw axis, this would not be appropriate. One component of $\boldsymbol{\rho}$ can be arbitrarily chosen, and any two of the component equations can then be solved to find the two other components of $\boldsymbol{\rho}$. All other points on the screw axis are then given by $\boldsymbol{\rho} + k\hat{\mathbf{w}}$, where k can take any value.

Table 1.4 contains the conversions between screw transformations and homogeneous transformations. Note that the equivalent rotation matrix for a screw transformation has the same form as the equivalent rotation matrix for an angle-axis representation of orientation in Table 1.1. Also, the auxiliary relationship that the vector $\boldsymbol{\rho}$ be orthogonal to the screw axis ($\hat{\mathbf{w}}^\top \boldsymbol{\rho} = 0$) is used in Table 1.4 to provide a unique conversion to the

Table 1.4 Conversions from a screw transformation to a homogeneous transformation and vice versa, with abbreviations $c_\theta := \cos \theta$, $s_\theta := \sin \theta$, and $v_\theta := 1 - \cos \theta$

Screw transformation to homogeneous transformation:

$${}^j\mathbf{R}_i = \begin{pmatrix} w_x^2 v_\theta + c_\theta & w_x w_y v_\theta - w_z s_\theta & w_x w_z v_\theta + w_y s_\theta \\ w_x w_y v_\theta + w_z s_\theta & w_y^2 v_\theta + c_\theta & w_y w_z v_\theta - w_x s_\theta \\ w_x w_z v_\theta - w_y s_\theta & w_y w_z v_\theta + w_x s_\theta & w_z^2 v_\theta + c_\theta \end{pmatrix}$$

$${}^j\mathbf{p}_i = (\mathbf{1}_{3 \times 3} - {}^j\mathbf{R}_i)\boldsymbol{\rho} + h\theta \hat{\mathbf{w}}$$

Homogeneous transformation to screw transformation:

$$\mathbf{l} = \begin{pmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{pmatrix}^\top$$

$$\theta = \text{sign}(\mathbf{l}^\top {}^j\mathbf{p}_i) \left| \cos^{-1} \left(\frac{r_{11} + r_{22} + r_{33} - 1}{2} \right) \right|$$

$$h = \frac{\mathbf{l}^\top {}^j\mathbf{p}_i}{2\theta \sin \theta}$$

$$\boldsymbol{\rho} = \frac{(\mathbf{1}_{3 \times 3} - {}^j\mathbf{R}_i^\top) {}^j\mathbf{p}_i}{2(1 - \cos \theta)}$$

$$\hat{\mathbf{w}} = \frac{\mathbf{l}}{2 \sin \theta}$$

screw transformation. The inverse result, that of finding the rotation matrix ${}^j\mathbf{R}_i$ and the translation ${}^j\mathbf{p}_i$ corresponding to a given screw displacement, is found from Rodrigues' equation.

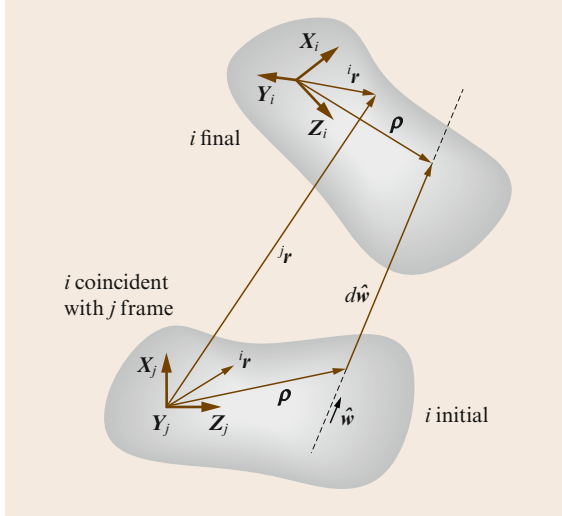


Fig. 1.1 Initial and final positions of an arbitrary point in a body undergoing a screw displacement; ${}^i\mathbf{r}$ is the position of the point relative to the moving frame, which is coincident with the fixed reference frame j in its initial position; ${}^j\mathbf{r}$ is the position of the point relative to the fixed frame after the screw displacement of the moving body

Rodrigues' Equation

Given a screw axis, the angular displacement of a body about it, and the translation of the body along it, the displacement of an arbitrary point in that body can be found. Viewing a matrix transformation as describing the displacement of the body, this is equivalent to finding the matrix transformation equivalent to a given screw displacement.

Referring to Fig. 1.1, the position vectors of a point before and after a screw displacement can be geometrically related as

$${}^j\mathbf{r} = {}^i\mathbf{r} + d\hat{\mathbf{w}} + \sin \theta \hat{\mathbf{w}} \times ({}^i\mathbf{r} - \boldsymbol{\rho}) - (1 - \cos \theta)({}^i\mathbf{r} - \boldsymbol{\rho}) - ({}^i\mathbf{r} - \boldsymbol{\rho}) \cdot \hat{\mathbf{w}} \hat{\mathbf{w}}, \quad (1.19)$$

where ${}^i\mathbf{r}$ and ${}^j\mathbf{r}$ denote the initial and final positions of the point, $\hat{\mathbf{w}}$ and $\boldsymbol{\rho}$ specify the screw axis, and θ and d give the displacement about it. This result is usually referred to as Rodrigues' equation [1.9], which can be written as a matrix transformation [1.10],

$${}^j\mathbf{r} = {}^j\mathbf{R}_i {}^i\mathbf{r} + {}^j\mathbf{p}_i, \quad (1.20)$$

since, when expanded, it gives three linear equations for the components of ${}^j\mathbf{r}$ in terms of those of ${}^i\mathbf{r}$.

$${}^j\mathbf{R}_i = \begin{pmatrix} w_x^2 v_\theta + c_\theta & w_x w_y v_\theta - w_z s_\theta & w_x w_z v_\theta + w_y s_\theta \\ w_x w_y v_\theta + w_z s_\theta & w_y^2 v_\theta + c_\theta & w_y w_z v_\theta - w_x s_\theta \\ w_x w_z v_\theta - w_y s_\theta & w_y w_z v_\theta + w_x s_\theta & w_z^2 v_\theta + c_\theta \end{pmatrix}$$

$${}^j\mathbf{p}_i = (\mathbf{1}_{3 \times 3} - {}^j\mathbf{R}_i)\boldsymbol{\rho} + h\theta \hat{\mathbf{w}},$$

where the abbreviations are $c_\theta := \cos \theta$, $s_\theta := \sin \theta$, and $v_\theta = 1 - \cos \theta$. The rotation matrix ${}^j\mathbf{R}_i$ expressed in this form is also called the screw matrix, and these equations give the elements of ${}^j\mathbf{R}_i$ and ${}^j\mathbf{p}_i$ in terms of the screw parameters.

An exception arises in the case of a pure translation, for which $\theta = 0$ and Rodrigues' equation becomes

$${}^j\mathbf{r} = {}^i\mathbf{r} + d\hat{\mathbf{w}}. \quad (1.21)$$

Substituting for this case, ${}^j\mathbf{R}_i = \mathbf{1}_{3 \times 3}$ and ${}^j\mathbf{p}_i = d\hat{\mathbf{w}}$.

Additional information on screw theory can be found in [1.11–15].

1.2.5 Matrix Exponential Parameterization

The position and orientation of a body can also be expressed in a unified fashion with an exponential mapping. This approach is introduced first with its application to pure rotation and expanded to rigid-body

motion. More details on the approach can be found in [1.16] and [1.17].

Exponential Coordinates for Rotation

The set of all orthogonal matrices with determinant 1, which is the set of all rotation matrices \mathbf{R} , is a *group* under the operation of matrix multiplication denoted as $SO(3) \subset \mathbb{R}^{3 \times 3}$ [1.18]. This stands for *special orthogonal* wherein *special* alludes to the $\det \mathbf{R}$ being +1 instead of ± 1 . The set of rotation matrices satisfies the four axioms of a group:

$$\begin{aligned} \text{closure:} & \quad \mathbf{R}_1 \mathbf{R}_2 \in SO(3) \quad \forall \mathbf{R}_1, \mathbf{R}_2 \in SO(3); \\ \text{identity:} & \quad \mathbf{1}_{3 \times 3} \mathbf{R} = \mathbf{R} \mathbf{1}_{3 \times 3} = \mathbf{R} \quad \forall \mathbf{R} \in SO(3); \\ \text{inverse:} & \quad \mathbf{R}^\top \in SO(3) \text{ is the unique inverse of } \mathbf{R} \\ & \quad \forall \mathbf{R} \in SO(3); \\ \text{associativity:} & \quad (\mathbf{R}_1 \mathbf{R}_2) \mathbf{R}_3 = \mathbf{R}_1 (\mathbf{R}_2 \mathbf{R}_3) \quad \forall \mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3 \\ & \quad \in SO(3). \end{aligned}$$

In the angle-axis representation presented in Sect. 1.2.2, orientation is expressed as an angle θ of rotation about an axis defined by the unit vector $\hat{\mathbf{w}}$. The equivalent rotation matrix found in Table 1.1 can be expressed as the exponential map

$$\begin{aligned} \mathbf{R} = e^{S(\hat{\mathbf{w}})\theta} &= \mathbf{1}_{3 \times 3} + \theta S(\hat{\mathbf{w}}) + \frac{\theta^2}{2!} S(\hat{\mathbf{w}})^2 \\ &+ \frac{\theta^3}{3!} S(\hat{\mathbf{w}})^3 + \dots, \end{aligned} \quad (1.22)$$

where $S(\hat{\mathbf{w}})$ is the unit skew-symmetric matrix

$$S(\hat{\mathbf{w}}) = \begin{pmatrix} 0 & -w_z & w_y \\ w_z & 0 & -w_x \\ -w_y & w_x & 0 \end{pmatrix}. \quad (1.23)$$

Thus, the exponential map transforms a skew-symmetric matrix $S(\hat{\mathbf{w}})$ that corresponds to an axis of rotation $\hat{\mathbf{w}}$ into an orthogonal matrix \mathbf{R} that corresponds to a rotation about the axis $\hat{\mathbf{w}}$ through an angle of θ . It can be shown that the closed-form expression for $e^{S(\hat{\mathbf{w}})\theta}$, which can be efficiently computed, is

$$e^{S(\hat{\mathbf{w}})\theta} = \mathbf{1}_{3 \times 3} + S(\hat{\mathbf{w}}) \sin \theta + S(\hat{\mathbf{w}})^2 (1 - \cos \theta). \quad (1.24)$$

The components of $(\theta w_x \theta w_y \theta w_z)^\top$, which are related to the elements of the rotation matrix \mathbf{R} in Table 1.2, are referred to as the *exponential coordinates* for \mathbf{R} .

Exponential Coordinates for Rigid-Body Motion

As indicated in Sect. 1.2.3, the position and orientation of a body can be expressed by the combination of a position vector $\mathbf{p} \in \mathbb{R}^3$ and a rotation matrix $\mathbf{R} \in SO(3)$.

The product space of \mathbb{R}^3 with $SO(3)$ is the group known as $SE(3)$, which stands for *special Euclidean*.

$$\begin{aligned} SE(3) &= \{(\mathbf{p}, \mathbf{R}) : \mathbf{p} \in \mathbb{R}^3, \mathbf{R} \in SO(3)\} \\ &= \mathbb{R}^3 \times SO(3). \end{aligned}$$

The set of homogeneous transformations satisfies the four axioms of a group:

- *closure*: $\mathbf{T}_1 \mathbf{T}_2 \in SE(3) \quad \forall \mathbf{T}_1, \mathbf{T}_2 \in SE(3)$;
- *identity*: $\mathbf{1}_{4 \times 4} \mathbf{T} = \mathbf{T} \mathbf{1}_{4 \times 4} = \mathbf{T} \quad \forall \mathbf{T} \in SE(3)$;
- *inverse*: the unique inverse of $\mathbf{T} \quad \forall \mathbf{T} \in SE(3)$ is given in (1.11);
- *associativity*: $(\mathbf{T}_1 \mathbf{T}_2) \mathbf{T}_3 = \mathbf{T}_1 (\mathbf{T}_2 \mathbf{T}_3) \quad \forall \mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3 \in SE(3)$.

In the screw transformation representation in Sect. 1.2.4, position and orientation are expressed by the angle θ of rotation about a screw axis defined by the unit vector $\hat{\mathbf{w}}$, the point \mathbf{p} on the axis such that $\hat{\mathbf{w}}^\top \mathbf{p} = 0$, and the pitch h of the screw axis. The equivalent homogeneous transformation found in Table 1.4 can be expressed as the exponential map

$$\mathbf{T} = e^{\hat{\xi}\theta} = \mathbf{1}_{4 \times 4} + \hat{\xi}\theta + \frac{(\hat{\xi}\theta)^2}{2!} + \frac{(\hat{\xi}\theta)^3}{3!} + \dots, \quad (1.25)$$

where

$$\hat{\xi} = \begin{pmatrix} S(\hat{\mathbf{w}}) & \mathbf{p} \\ \mathbf{0}^\top & 0 \end{pmatrix} \quad (1.26)$$

is the generalization of the unit skew-symmetric matrix $S(\hat{\mathbf{w}})$ known as a *twist*. The twist coordinates of $\hat{\xi}$ are given by $\xi := (\hat{\mathbf{w}}^\top \mathbf{p})^\top$. It can be shown that the closed-form expression for $e^{\hat{\xi}\theta}$ is

$$e^{\hat{\xi}\theta} = \begin{pmatrix} e^{S(\hat{\mathbf{w}})\theta} & (\mathbf{1}_{3 \times 3} - e^{S(\hat{\mathbf{w}})\theta})(\hat{\mathbf{w}} \times \mathbf{p}) + \hat{\mathbf{w}}^\top \mathbf{p} \theta \hat{\mathbf{w}} \\ \mathbf{0}^\top & 1 \end{pmatrix}. \quad (1.27)$$

Comparison of this result with the conversion between homogeneous and screw transformations in Table 1.4 yields

$$\mathbf{v} = \mathbf{p} \times \hat{\mathbf{w}} \quad (1.28)$$

and

$$h = \hat{\mathbf{w}}^\top \mathbf{p}. \quad (1.29)$$

Thus, the exponential map for a twist transforms the initial pose of a body into its final pose. It gives the relative rigid-body motion. The vector $\xi\theta$ contains the *exponential coordinates* for the rigid-body transformation.

As for screw transformations, the case of pure translation is unique. In this case, $\hat{\mathbf{w}} = \mathbf{0}$, so

$$\mathbf{e}^{\hat{\xi}\theta} = \begin{pmatrix} \mathbf{1}_{3 \times 3} & \theta \mathbf{v} \\ \mathbf{0}^\top & 1 \end{pmatrix}. \quad (1.30)$$

1.2.6 Plücker Coordinates

A minimum of four coordinates are needed to define a line in space. The Plücker coordinates of a line form a six-dimensional vector, so they are superabundant by two. They can be viewed as a pair of three-dimensional vectors; one is parallel to the line, and the other is the *moment* of that vector about the origin. Thus, if \mathbf{u} is any vector parallel to the line and $\boldsymbol{\rho}$ is the position of any point on the line relative to the origin, the Plücker coordinates (L, M, N, P, Q , and R) are given by

$$(L, M, N) = \mathbf{u}^\top; (P, Q, R) = (\boldsymbol{\rho} \times \mathbf{u})^\top. \quad (1.31)$$

For simply defining a line, the magnitude of \mathbf{u} is not unique, nor is the component of $\boldsymbol{\rho}$ parallel to \mathbf{u} . Two auxiliary relationships are imposed to reduce the set to just four independent coordinates. One is that the scalar product of the two three-dimensional vectors is identically zero.

$$LP + MQ + NR \equiv 0. \quad (1.32)$$

The other is the invariance of the line designated when the coordinates are all multiplied by the same scaling factor.

$$(L, M, N, P, Q, R) \equiv (kL, kM, kN, kP, kQ, kR). \quad (1.33)$$

This relationship may take the form of constraining \mathbf{u} to have unit magnitude so that L, M , and N are the direction cosines.

In this handbook, it is often useful to express velocities in Plücker coordinates, wherein unlike the definition of lines, the magnitudes of the two three-dimensional vectors are not arbitrary. This leads to the motor notation of *von Mises* [1.9, 19] and *Everett* [1.20]. For instantaneously coincident coordinate frames, one fixed and the

other embedded in the moving body, $\boldsymbol{\omega}$ is the angular velocity of the body and \mathbf{v}_O is the velocity of the origin O of the body-fixed frame when both are expressed relative to the fixed frame. This provides a Plücker coordinate system for the spatial velocity \mathbf{v} of the body. The Plücker coordinates of \mathbf{v} are simply the Cartesian coordinates of $\boldsymbol{\omega}$ and \mathbf{v}_O ,

$$\mathbf{v} = \begin{pmatrix} \boldsymbol{\omega} \\ \mathbf{v} \end{pmatrix}. \quad (1.34)$$

The transformation from Plücker coordinate system i to Plücker coordinate system j for spatial velocities is achieved with the spatial transform jX_i . If \mathbf{v}_i and \mathbf{v}_j denote the spatial velocities of a body relative to the i and j frames, respectively, and ${}^j\mathbf{p}_i$ and ${}^j\mathbf{R}_i$ denote the position and orientation of frame i relative to frame j ,

$$\mathbf{v}_j = {}^jX_i \mathbf{v}_i, \quad (1.35)$$

where

$${}^jX_i = \begin{pmatrix} {}^j\mathbf{R}_i & \mathbf{0}_{3 \times 3} \\ S({}^j\mathbf{p}_i){}^j\mathbf{R}_i & {}^j\mathbf{R}_i \end{pmatrix}, \quad (1.36)$$

such that

$${}^jX_i^{-1} = {}^iX_j = \begin{pmatrix} {}^i\mathbf{R}_j & \mathbf{0}_{3 \times 3} \\ -{}^i\mathbf{R}_j S({}^j\mathbf{p}_i) & {}^i\mathbf{R}_j \end{pmatrix} \quad (1.37)$$

and

$${}^kX_i = {}^kX_j {}^jX_i, \quad (1.38)$$

and $S({}^j\mathbf{p}_i)$ is the skew-symmetric matrix

$$\begin{pmatrix} 0 & -{}^j p_i^z & {}^j p_i^y \\ {}^j p_i^z & 0 & -{}^j p_i^x \\ -{}^j p_i^y & {}^j p_i^x & 0 \end{pmatrix}. \quad (1.39)$$

Spatial vector notation, which includes the spatial velocities and transforms briefly mentioned here, is treated in greater depth in Sect. 2.2. Specifically, Table 2.1 gives a computationally efficient algorithm for applying a spatial transform.

1.3 Joint Kinematics

Unless explicitly stated otherwise, the kinematic description of robotic mechanisms typically employs a number of idealizations. The links that compose the robotic mechanism are assumed to be perfectly rigid

bodies having surfaces that are geometrically perfect in both position and shape. Accordingly, these rigid bodies are connected together at joints where their idealized surfaces are in ideal contact without any

clearance between them. The respective geometries of these surfaces in contact determine the freedom of motion between the two links, or the *joint kinematics*.

A kinematic joint is a connection between two bodies that constrains their relative motion. Two bodies that are in contact with one another create a simple kinematic joint. The surfaces of the two bodies that are in contact are able to move over one another, thereby permitting relative motion of the two bodies. Simple kinematic joints are classified as lower pair joints if contact occurs over surfaces [1.22] and as higher pair joints if contact occurs only at points or along lines.

A joint model describes the motion of a frame fixed in one body of a joint relative to a frame fixed in the other body. The motion is expressed as a function of the joint's motion variables, and other elements of a joint model include the rotation matrix, position vector, free modes, and constrained modes. The free modes of a joint define the directions in which motion is allowed. They are represented by the $6 \times n_i$ matrix Φ_i whose columns are the Plücker coordinates of the allowable motion. This

matrix relates the spatial velocity vector across the joint \mathbf{v}_{rel} to the joint velocity vector $\dot{\mathbf{q}}$,

$$\mathbf{v}_{\text{rel}} = \Phi_i \dot{\mathbf{q}}. \quad (1.40)$$

In contrast, the constrained modes of a joint define the directions in which motion is not allowed. They are represented by the $6 \times (6 - n_i)$ matrix Φ_i^c that is complementary to Φ_i . Tables 1.5 and 1.6 contain the formulas of the joint models for all of the joints described in this section. They are used extensively for the dynamic analysis presented in Chap. 2. Additional information on joints can be found in Chap. 3.

1.3.1 Lower Pair Joints

Lower pair joints are mechanically attractive since wear is spread over the whole surface and lubricant is trapped in the small clearance space (in nonidealized systems) between the surfaces, resulting in relatively good lubrication. As can be proved [1.23] from the requirement for surface contact, there are only six possible forms of lower pair: revolute, prismatic, helical, cylindrical, spherical, and planar joints.

Table 1.5 Joint model formulas for one-degree-of-freedom lower pair joints, with abbreviations $c_{\theta_i} := \cos \theta_i$ and $s_{\theta_i} := \sin \theta_i$. (Adapted in part from Table 4.1 in [1.21])

Joint type	Joint rotation matrix ${}^j\mathbf{R}_i$	Position vector ${}^j\mathbf{p}_i$	Free modes Φ_i	Constrained modes Φ_i^c	Pose state vars.	$\dot{\mathbf{q}}_i$
Revolute R	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	θ_i	$\dot{\theta}_i$
Prismatic P	$\mathbf{1}_{3 \times 3}$	$\begin{pmatrix} 0 \\ 0 \\ d_i \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$	d_i	\dot{d}_i
Helical H (pitch h)	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ h\theta_i \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ h \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -h \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	θ_i	$\dot{\theta}_i$

Table 1.6 Joint model formulas for higher-degree-of-freedom lower pair joints, universal joint, rolling contact joint, and 6-DOF joint, with abbreviations $c_{\theta_i} := \cos \theta_i$ and $s_{\theta_i} := \sin \theta_i$. (Adapted in part from Table 4.1 in [1.21]) *The Euler angles α_i , β_i , and γ_i could be used in place of the unit quaternion ϵ_i to represent orientation

Joint type	Joint rotation matrix jR_i	Position vector jp_i	Free modes Φ_i	Constrained modes Φ_i^c	Pose Variables	Velocity Variables \dot{q}_i
Cylindrical C	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ d_i \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	θ_i d_i	$\begin{pmatrix} \dot{\theta}_i \\ \dot{d}_i \end{pmatrix}$
Spherical* S	$\begin{pmatrix} \text{see} \\ \text{Table 1.1} \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	ϵ_i	$\omega_i \text{ rel}$
Planar	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} c_{\theta_i} d_{xi} - s_{\theta_i} d_{yi} \\ s_{\theta_i} d_{xi} + c_{\theta_i} d_{yi} \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	θ_i d_{xi} d_{yi}	$\begin{pmatrix} \dot{\theta}_i \\ \dot{d}_{xi} \\ \dot{d}_{yi} \end{pmatrix}$
Flat planar rolling contact (fixed radius r)	$\begin{pmatrix} c_{\theta_i} & -s_{\theta_i} & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} r\theta_i c_{\theta_i} - rs_{\theta_i} \\ -r\theta_i s_{\theta_i} - rc_{\theta_i} \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 1 \\ r \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -r & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	θ_i	$\dot{\theta}_i$
Universal U	$\begin{pmatrix} c_{\alpha_i} c_{\beta_i} & -s_{\alpha_i} & c_{\alpha_i} s_{\beta_i} \\ s_{\alpha_i} c_{\beta_i} & c_{\alpha_i} & s_{\alpha_i} s_{\beta_i} \\ -s_{\beta_i} & 0 & c_{\beta_i} \end{pmatrix}$	$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$	$\begin{pmatrix} -s_{\beta_i} & 0 \\ 0 & 1 \\ c_{\beta_i} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$	$\begin{pmatrix} c_{\beta_i} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ s_{\beta_i} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	α_i β_i	$\begin{pmatrix} \dot{\alpha}_i \\ \dot{\beta}_i \end{pmatrix}$
6-DOF*	$\begin{pmatrix} \text{see} \\ \text{Table 1.1} \end{pmatrix}$	0p_i	$\mathbf{1}_{6 \times 6}$		ϵ_i 0p_i	$\begin{pmatrix} \omega_i \\ v_i \end{pmatrix}$

Revolute

The most general form of a revolute joint, often abbreviated as “R” and sometimes referred to colloquially as a hinge or pin joint, is a lower pair composed of two congruent surfaces of revolution. The surfaces are the same except one of them is an external surface, convex in any plane normal to the axis of revolution, and one is an internal surface, concave in any plane normal to

the axis. The surfaces may not be solely in the form of right circular cylinders, since surfaces of that form do not provide any constraint on axial sliding. A revolute joint permits only rotation of one of the bodies joined relative to the other. The position of one body relative to the other may be expressed as the angle between two lines normal to the joint axis, one fixed in each body. Thus, the joint has one *degree of freedom* (DOF). When

the \hat{z} axis of coordinate frame i is aligned with a revolute joint axis, the formulas in Table 1.5 define the revolute joint model.

Prismatic

The most general form of a prismatic joint, often abbreviated as “P” and sometimes referred to colloquially as a sliding joint, is a lower pair formed from two congruent general cylindrical surfaces. These may not be right circular cylindrical surfaces. A general cylindrical surface is obtained by extruding any curve in a constant direction. Again, one surface is internal and the other is an external surface. A prismatic joint permits only sliding of one of the members joined relative to the other along the direction of extrusion. The position of one body relative to the other is determined by the distance between two points on a line parallel to the direction of sliding, with one point fixed in each body. Thus, this joint also has one degree of freedom. When the \hat{z} axis of coordinate frame i is aligned with a prismatic joint axis, the formulas in Table 1.5 define the prismatic joint model.

Helical

The most general form of a helical joint, often abbreviated as “H” and sometimes referred to colloquially as a screw joint, is a lower pair formed from two helicoidal surfaces formed by extruding any curve along a helical path. The simple example is a screw and nut wherein the basic generating curve is a pair of straight lines. The angle of rotation about the axis of the screw joint θ is directly related to the distance of displacement of one body relative to the other along that axis d by the expression $d = h\theta$, where the constant h is called the pitch of the screw. When the \hat{z} axis of coordinate frame i is aligned with a helical joint axis, the formulas in Table 1.5 define the helical joint model.

Cylindrical

A cylindrical joint, often abbreviated as “C”, is a lower pair formed by contact of two congruent right circular cylinders, one an internal surface and the other an external surface. It permits both rotation about the cylinder axis and sliding parallel to it. Therefore, it is a joint with two degrees of freedom. Lower pair joints with more than one degree of freedom are easily replaced by kinematically equivalent compound joints (see Sect. 1.3.3) that are serial chains of one-degree-of-freedom lower pairs. In the present case, the cylindrical joint can be replaced by a revolute in series with a prismatic joint whose direction of sliding is parallel to the revolute

axis. While simpler to implement using the geometric representation discussed in Sect. 1.4, this approach has disadvantages for dynamic simulation. Modeling a single cylindrical joint as a combination of a prismatic and revolute joint requires the addition of a virtual link between the two with zero mass and zero length. The massless link can create computational problems. When the \hat{z} axis of coordinate frame i is aligned with a cylindrical joint axis, the formulas in Table 1.6 define the cylindrical joint model.

Spherical

A spherical joint, often abbreviated as “S”, is a lower pair formed by contact of two congruent spherical surfaces. Once again, one is an internal surface, and the other is an external surface. A spherical joint permits rotation about any line through the center of the sphere. Thus, it permits independent rotation about axes in up to three different directions and has three degrees of freedom. A spherical joint is easily replaced by a kinematically equivalent compound joint consisting of three revolutes that have concurrent axes. They do not need to be successively orthogonal, but often they are implemented that way. The arrangement is, in general, kinematically equivalent to a spherical joint, but it does exhibit a singularity when the revolute joint axes become coplanar. This is as compared to the native spherical joint that never has such a singularity. Likewise, if a spherical joint is modeled in simulation as three revolutes, computational difficulties again can arise from the necessary inclusion of massless virtual links having zero length. The joint model formulas of a spherical joint are given in Table 1.6.

Planar

A planar joint is formed by planar contacting surfaces. Like the spherical joint, it is a lower pair joint with three degrees of freedom. A kinematically equivalent compound joint consisting of a serial chain of three revolutes with parallel axes can replace a planar joint. As was the case with the spherical joint, the compound joint exhibits a singularity when the revolute axes become coplanar. When the \hat{z} axis of coordinate frame i is aligned with the normal to the plane of contact, the formulas in Table 1.6 define the planar joint model.

1.3.2 Higher Pair Joints

Some higher pair joints also have attractive properties, particularly rolling pairs in which one body rolls without slipping over the surface of the other. This is mechan-

ically attractive since the absence of sliding means the absence of abrasive wear. However, since ideal contact occurs at a point, or along a line, application of a load across the joint may lead to very high local stresses resulting in other forms of material failure and, hence, wear. Higher pair joints can be used to create kinematic joints with special geometric properties, as in the case of a gear pair, or a cam and follower pair.

Rolling Contact

Rolling contact actually encompasses several different geometries. Rolling contact in planar motion permits one degree of freedom of relative motion as in the case of a roller bearing, for example. As noted above, rolling contact has desirable wear properties since the absence of sliding means the absence of abrasive wear. Planar rolling contact can take place along a line, thereby spreading the load and wear somewhat. Three-dimensional rolling contact allows rotation about any axis through the point of contact that is, in principle, unique. Hence, a three-dimensional rolling contact pair permits relative motion with three degrees of freedom. When the \hat{z} axis of coordinate frame i is aligned with the axis of rotation and passes through the center of the roller of fixed radius r , the formulas in Table 1.6 define the planar rolling contact joint model for a roller on a flat surface.

Regardless of whether the joint is planar or three-dimensional, the *no-slip* condition associated with a rolling contact joint requires that the instantaneous relative velocity between the points on the two bodies in contact be zero. If P is the point of rolling contact between bodies i and j ,

$$\mathbf{v}_{P_i/P_j} = 0. \quad (1.41)$$

Likewise, relative acceleration is in the direction of the common normal to the two surfaces at the point of contact. Because the constraint associated with the joint is expressed in terms of velocity, it is nonholonomic, as discussed in Sect. 1.3.6. A more detailed discussion of the kinematic constraints for rolling contact is found in Sect. 17.2.2 of Chap. 17.

1.3.3 Compound Joints

Compound kinematic joints are connections between two bodies formed by chains of other members and simple kinematic joints. A compound joint may constrain the relative motion of the two bodies joined in the same way as a simple joint. In such a case, the two joints are said to be kinematically equivalent.

Universal

A universal joint, often abbreviated as “U” and referred to as a Cardan or Hooke joint, is a compound joint with two degrees of freedom. It consists of a serial chain of two revolute whose axes intersect orthogonally. The joint model for a universal joint, in which, from Euler angle notation, α_i is the first rotation about the Z -axis and then β_i is the rotation about the Y -axis, is given in Table 1.6. This is a joint for which the matrices Φ_i and Φ_i^c are not constant, so in general, $\dot{\Phi}_i \neq 0$ and $\dot{\Phi}_i^c \neq 0$. In this case, the orientation of the outboard reference frame varies with α_i .

1.3.4 6-DOF Joint

The motion of two bodies not jointed together can be modeled as a six-degree-of-freedom *joint* that introduces no constraints. This is particularly useful for mobile robots, such as aircraft, that make at most intermittent contact with the ground, and thus, a body in free motion relative to the fixed frame is termed a *floating base*. Such a free motion joint model enables the position and orientation of a floating base in space to be expressed with six joint variables. The 6-DOF joint model is included in Table 1.6.

1.3.5 Physical Realization

In an actual robotic mechanism, the joints may have physical limits beyond which motion is prohibited. The workspace (see Sect. 1.5) of a robotic manipulator is determined by considering the combined limits and freedom of motion of all of the joints within the mechanism. Revolute joints are easily actuated by rotating motors and are, therefore, extremely common in robotic systems. They may also be present as passive, unactuated joints. Also common, although less so than revolute, prismatic joints are relatively easily actuated by means of linear actuators such as hydraulic or pneumatic cylinders, ball screws, or screw jacks. They always have motion limits since unidirectional sliding can, in principle, produce infinite displacements. Screw joints are most often found in robotic mechanisms as constituents of linear actuators such as screw jacks and ball screws and are seldom used as primary kinematic joints. Joints with more than one degree of freedom are generally used passively in robotic mechanisms because each degree of freedom of an active joint must be separately actuated. Passive spherical joints are quite often found in robotic mechanisms, while passive planar joints are only occasionally found. The effect of an actuated spherical joint

is achieved by employing the kinematically equivalent combination of three revolute and actuating each. Universal joints are used in robotic mechanisms in both active and passive forms.

Serial chains are commonly denoted by the abbreviations for the joints they contain in the order in which they appear in the chain. For example, an RPR chain contains three links, the first jointed to the base with a revolute and to the second with a prismatic, while the second and third are jointed together with another revolute. If all of the joints are identical, the notation consists of the number of joints preceding the joint abbreviation, such as 6R for a six-axis serial-chain manipulator containing only revolute joints.

Joints are realized with hardware that is more complex than the idealizations presented in Sects. 1.3.1 and 1.3.2. For example, a revolute joint may be achieved with a ball bearing composed of a set of bearing balls trapped between two journals. The balls ideally roll without slipping on the journals, thereby taking advantage of the special properties of rolling contact joints. A prismatic joint may be realized by means of a roller-rail assembly.

1.3.6 Holonomic and Nonholonomic Constraints

With the exception of rolling contact, all of the constraints associated with the joints discussed in the preceding sections can be expressed mathematically by equations containing only the joint position variables.

These are called holonomic constraints. The number of equations, and hence the number of constraints, is $6 - n$, where n is the number of degrees of freedom of the joint. The constraints are intrinsically part of the axial joint model.

A nonholonomic constraint is one that cannot be expressed in terms of the position variables alone, but includes the time derivative of one or more of those variables. These constraint equations cannot be integrated to obtain relationships solely between the joint variables. The most common example in robotic systems arises from the use of a wheel or roller that rolls without slipping on another member. Nonholonomic constraints, particularly as they apply to wheeled robots, are discussed in more detail in Chap. 17.

1.3.7 Generalized Coordinates

In a robot manipulator consisting of N bodies, $6N$ coordinates are required to specify the position and orientation of all the bodies relative to a coordinate frame. Since some of those bodies are jointed together, a number of constraint equations will establish relationships between some of these coordinates. In this case, the $6N$ coordinates can be expressed as functions of a smaller set of coordinates \mathbf{q} that are all independent. The coordinates in this set are known as generalized coordinates, and motions associated with these coordinates are consistent with all of the constraints. The joint variables \mathbf{q} of a robot manipulator form a set of generalized coordinates [1.24, 25].

1.4 Geometric Representation

The geometry of a robotic manipulator is conveniently defined by attaching reference frames to each link. While these frames could be located arbitrarily, it is advantageous both for consistency and computational efficiency to adhere to a convention for locating the frames on the links. *Denavit and Hartenberg* [1.26] introduced the foundational convention that has been adapted in a number of different ways, one of which is the convention introduced by *Khalil and Dombre* [1.27] used throughout this handbook. In all of its forms, the convention requires only four rather than six parameters to locate one reference frame relative to another. The four parameters are the link length a_i , the link twist α_i , the joint offset d_i , and the joint angle θ_i . This parsimony is achieved through judicious placement of the reference

frame origins and axes such that the $\hat{\mathbf{x}}$ axis of one frame both intersects and is perpendicular to the $\hat{\mathbf{z}}$ axis of the preceding reference frame. The convention is applicable to manipulators consisting of revolute and prismatic joints, so when multiple-degree-of-freedom joints are present, they are modeled as combinations of revolute and prismatic joints, as discussed in Sect. 1.3.

There are essentially four different forms of the convention for locating reference frames in a robotic mechanism. Each exhibits its own advantages by managing trade-offs of intuitive presentation. In the original *Denavit and Hartenberg* [1.26] convention, joint i is located between links i and $i + 1$, so it is on the outboard side of link i . Also, the joint offset d_i and joint angle θ_i are measured along and about the $i - 1$ joint axis, so the

subscripts of the joint parameters do not match that of the joint axis. *Waldron* [1.28] and *Paul* [1.29] modified the labeling of axes in the original convention such that joint i is located between links $i - 1$ and i in order to make it consistent with the base member of a serial chain being member 0. This places joint i at the inboard side of link i and is the convention used in all of the other modified versions. Furthermore, Waldron and Paul addressed the mismatch between subscripts of the joint parameters and joint axes by placing the \hat{z}_i axis along the $i + 1$ joint axis. This, of course, relocates the subscript mismatch to the correspondence between the joint axis and the \hat{z} axis of the reference frame. *Craig* [1.30] eliminated all of the subscript mismatches by placing the \hat{z}_i axis along joint i , but at the expense of the homogeneous transformation ${}^{i-1}T_i$ being formed with a mixture of joint parameters with subscript i and link parameters with subscript $i - 1$. *Khalil* and *Dombre* [1.27] introduced another variation similar to Craig's except that it defines the link parameters a_i and α_i along and about the \hat{x}_{i-1} axis. In this case, the homogeneous transformation ${}^{i-1}T_i$ is formed only with parameters with subscript i , and the subscript mismatch is such that a_i and α_i indicate the length and twist of link $i - 1$ rather than link i . Thus, in summary, the advantages of the convention used throughout this handbook compared to the alternative conventions are

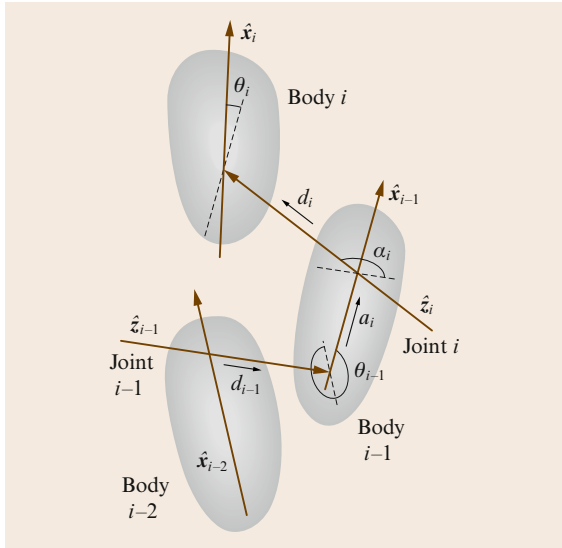


Fig. 1.2 Schematic of the numbering of bodies and joints in a robotic manipulator, the convention for attaching reference frames to the bodies, and the definitions of the four parameters, a_i , α_i , d_i , and θ_i , that locate one frame relative to another

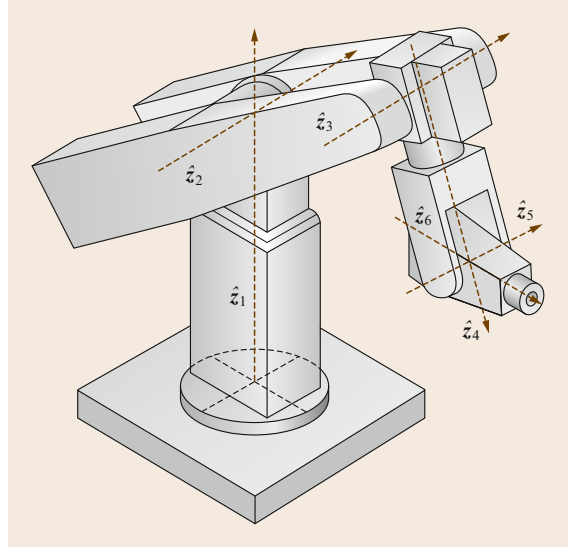


Fig. 1.3 Example six-degree-of-freedom serial chain manipulator composed of an articulated arm with no joint offsets and a spherical wrist

that the \hat{z} axes of the reference frames share the common subscript of the joint axes, and the four parameters that define the spatial transform from reference frame i to reference frame $i - 1$ all share the common subscript i .

In this handbook, the convention for serial chain manipulators is shown in Fig. 1.2 and summarized as follows. The numbering of bodies and joints follows the convention:

- the N moving bodies of the robotic mechanism are numbered from 1 to N . The number of the base is 0.
- the N joints of the robotic mechanism are numbered from 1 to N , with joint i located between members $i - 1$ and i .

With this numbering scheme, the attachment of reference frames follows the convention:

- the \hat{z}_i axis is located along the axis of joint i ,
- the \hat{x}_{i-1} axis is located along the common normal between the \hat{z}_{i-1} and \hat{z}_i axes.

Using the attached frames, the four parameters that locate one frame relative to another are defined as

- a_i is the distance from \hat{z}_{i-1} to \hat{z}_i along \hat{x}_{i-1} ,
- α_i is the angle from \hat{z}_{i-1} to \hat{z}_i about \hat{x}_{i-1} ,
- d_i is the distance from \hat{x}_{i-1} to \hat{x}_i along \hat{z}_i ,
- θ_i is the angle from \hat{x}_{i-1} to \hat{x}_i about \hat{z}_i .

Table 1.7 Geometric parameters of the example serial chain manipulator in Fig. 1.3

i	α_i	a_i	d_i	θ_i
1	0	0	0	θ_1
2	$-\frac{\pi}{2}$	0	0	θ_2
3	0	a_3	0	θ_3
4	$-\frac{\pi}{2}$	0	d_4	θ_4
5	$\frac{\pi}{2}$	0	0	θ_5
6	$-\frac{\pi}{2}$	0	0	θ_6

The geometric parameters for the example manipulator shown in Fig. 1.3 are listed in Table 1.7. All of the joints of this manipulator are revolute, and joint 1 has a vertical orientation. Joint 2 is perpendicular to joint 1 and intersects it. Joint 3 is parallel to joint 2, and the length of link 2 is a_3 . Joint 4 is perpendicular to joint 3 and intersects it. Joint 5 likewise intersects joint 4 perpendicularly at an offset of d_4 from joint 3. Finally, joint 6 intersects joint 5 perpendicularly.

1.5 Workspace

Most generally, the workspace of a robotic manipulator is the total volume swept out by the end-effector as the manipulator executes all possible motions. The workspace is determined by the geometry of the manipulator and the limits of the joint motions. It is more specific to define the reachable workspace as the total locus of points at which the end-effector can be placed and the dextrous workspace [1.31] as the subset of those points at which the end-effector can be placed while having an arbitrary orientation. Dexterous workspaces exist only for certain idealized geometries, so real industrial manipulators with joint motion limits almost never possess dextrous workspaces.

Many serial-chain robotic manipulators are designed such that their joints can be partitioned into a regional structure and an orientation structure. The joints in the regional structure accomplish the positioning of the end-effector in space, and the joints in the orientation structure accomplish the orientation of the end-effector. Typically, the inboard joints of a serial chain manipulator comprise the regional structure, while the outboard joints comprise the orientation structure. Also, since prismatic joints provide no capability for rotation, they are generally not employed within the orientation structure.

With this convention, reference frame i can be located relative to reference frame $i-1$ by executing a rotation through an angle α_i about the \hat{x}_{i-1} axis, a translation of distance a_i along \hat{x}_{i-1} , a rotation through an angle θ_i about the \hat{z}_i axis, and a translation of distance d_i along \hat{z}_i . Through concatenation of these individual transformations,

$$\text{Rot}(\hat{x}_{i-1}, \alpha_i) \text{Trans}(\hat{x}_{i-1}, a_i) \text{Rot}(\hat{z}_i, \theta_i)$$

$$\text{Trans}(\hat{z}_i, d_i),$$

the equivalent homogeneous transformation is,

$${}^{i-1}T_i = \begin{pmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_i \\ \sin \theta_i \cos \alpha_i & \cos \theta_i \cos \alpha_i & -\sin \alpha_i & -\sin \alpha_i d_i \\ \sin \theta_i \sin \alpha_i & \cos \theta_i \sin \alpha_i & \cos \alpha_i & \cos \alpha_i d_i \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (1.42)$$

The identification of geometric parameters is addressed in Chap. 14.

The regional workspace volume can be calculated from the known geometry of the serial-chain manipulator and motion limits of the joints. With three inboard joints comprising the regional structure, the area of workspace for the outer two (joints 2 and 3) is computed first, and then the volume is calculated by integrating over the joint variable of the remaining inboard joint (joint 1). In the case of a prismatic joint, this simply involves multiplying the area by the total length of travel of the prismatic joint. In the more common case of a revolute joint, it involves rotating the area about the joint axis through the full range of motion of the revolute [1.32]. By the theorem of Pappus, the associated volume V is

$$V = A \bar{r} \gamma, \quad (1.43)$$

where A is the area, \bar{r} is the distance from the area's centroid to the axis, and γ is the angle through which the area is rotated. The boundaries of the area are determined by tracing the motion of a reference point in the end-effector, typically the center of rotation of the wrist that serves as the orientation structure. Starting with each of the two joints at motion limits and with joint 2 locked, joint 3 is moved until its second motion limit is reached. Joint 3 is then locked, and joint 2 is freed to move to its second motion limit. Joint 2 is

again locked, while joint 3 is freed to move back to its original motion limit. Finally, joint 3 is locked, and joint 2 freed to move likewise to its original motion limit. In this way, the trace of the reference point is

1.6 Forward Kinematics

The forward kinematics problem for a serial-chain manipulator is to find the position and orientation of the end-effector relative to the base given the positions of all of the joints and the values of all of the geometric link parameters. Often, a frame fixed in the end-effector is referred to as the *tool frame*, and while fixed in the final link N , it in general has a constant offset in both position and orientation from frame N . Likewise, a *station frame* is often located in the base to establish the location of the task to be performed. This frame generally has a constant offset in its pose relative to frame 0, which is also fixed in the base.

A more general expression of the forward kinematics problem is to find the relative position and orientation of any two designated members given the geometric structure of the manipulator and the values of a number of joint positions equal to the number of degrees of freedom of the mechanism. The forward kinematics problem is critical for developing manipulator coordination algorithms because joint positions are typically measured by sensors mounted on the joints and it is necessary to calculate the positions of the joint axes relative to the fixed reference frame.

In practice, the forward kinematics problem is solved by calculating the transformation between a reference frame fixed in the end-effector and another reference frame fixed in the base, i.e., between the tool and station frames. This is straightforward for a serial chain since the transformation describing the position of the end-effector relative to the base is obtained by simply concatenating transformations between frames fixed in adjacent links of the chain. The convention for the geometric representation of a manipulator presented in Sect. 1.4 reduces this to finding an equivalent 4×4 homogeneous transformation matrix that relates the spatial displacement of the end-effector reference frame to the base reference frame.

For the example serial-chain manipulator shown in Fig. 1.3 and neglecting the addition of tool and station frames, the transformation is

$${}^0T_6 = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 {}^4T_5 {}^5T_6. \quad (1.44)$$

a closed curve whose area and centroid can be calculated mathematically.

More details on manipulator workspace can be found in Chaps. 3 and 10.

Table 1.8 contains the elements of 0T_6 that are calculated using Table 1.7 and (1.42).

Once again, homogeneous transformations provide a compact notation, but are computationally inefficient for solving the forward kinematics problem. A reduction in computation can be achieved by separating the

Table 1.8 Forward kinematics of the example serial chain manipulator in Fig. 1.3, with abbreviations $c_{\theta_i} := \cos \theta_i$ and $s_{\theta_i} := \sin \theta_i$

$${}^0T_6 = \begin{pmatrix} r_{11} & r_{12} & r_{13} & {}^0p_6^x \\ r_{21} & r_{22} & r_{23} & {}^0p_6^y \\ r_{31} & r_{32} & r_{33} & {}^0p_6^z \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$\begin{aligned} r_{11} &= c_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(s_{\theta_4}s_{\theta_6} - c_{\theta_4}c_{\theta_5}c_{\theta_6}) \\ &\quad - c_{\theta_1}s_{\theta_5}c_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\ &\quad + s_{\theta_1}(s_{\theta_4}c_{\theta_5}c_{\theta_6} + c_{\theta_4}s_{\theta_6}), \\ r_{21} &= s_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(s_{\theta_4}s_{\theta_6} - c_{\theta_4}c_{\theta_5}c_{\theta_6}) \\ &\quad - s_{\theta_1}s_{\theta_5}c_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\ &\quad - c_{\theta_1}(s_{\theta_4}c_{\theta_5}c_{\theta_6} + c_{\theta_4}s_{\theta_6}), \\ r_{31} &= (c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3})(s_{\theta_4}s_{\theta_6} - c_{\theta_4}c_{\theta_5}c_{\theta_6}) \\ &\quad + s_{\theta_5}c_{\theta_6}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}), \\ r_{12} &= c_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(c_{\theta_4}c_{\theta_5}s_{\theta_6} + s_{\theta_4}c_{\theta_6}) \\ &\quad + c_{\theta_1}s_{\theta_5}s_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\ &\quad + s_{\theta_1}(c_{\theta_4}c_{\theta_6} - s_{\theta_4}c_{\theta_5}s_{\theta_6}), \\ r_{22} &= s_{\theta_1}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3})(c_{\theta_4}c_{\theta_5}s_{\theta_6} + s_{\theta_4}c_{\theta_6}) \\ &\quad + s_{\theta_1}s_{\theta_5}s_{\theta_6}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\ &\quad - c_{\theta_1}(c_{\theta_4}c_{\theta_6} - s_{\theta_4}c_{\theta_5}s_{\theta_6}), \\ r_{32} &= (c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3})(c_{\theta_4}c_{\theta_5}s_{\theta_6} + s_{\theta_4}c_{\theta_6}) \\ &\quad - s_{\theta_5}s_{\theta_6}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}), \\ r_{13} &= c_{\theta_1}c_{\theta_4}s_{\theta_5}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}) \\ &\quad - c_{\theta_1}c_{\theta_5}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\ &\quad - s_{\theta_1}s_{\theta_4}s_{\theta_5}, \\ r_{23} &= s_{\theta_1}c_{\theta_4}s_{\theta_5}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}) \\ &\quad - s_{\theta_1}c_{\theta_5}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) + c_{\theta_1}s_{\theta_4}s_{\theta_5}, \\ r_{33} &= c_{\theta_4}s_{\theta_5}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}) \\ &\quad + c_{\theta_5}(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}), \\ {}^0p_6^x &= a_3c_{\theta_1}c_{\theta_2} - d_4c_{\theta_1}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}), \\ {}^0p_6^y &= a_3s_{\theta_1}c_{\theta_2} - d_4s_{\theta_1}(c_{\theta_2}s_{\theta_3} + s_{\theta_2}c_{\theta_3}), \\ {}^0p_6^z &= -a_3s_{\theta_2} + d_4(s_{\theta_2}s_{\theta_3} - c_{\theta_2}c_{\theta_3}). \end{aligned}$$

position and orientation portions of the transformation to eliminate all multiplications by the 0 and 1 elements of the matrices. In Chap. 2, calculations are made using the spatial vector notation briefly introduced here in Sect. 1.2.6 and explained in detail in Sect. 2.2. This approach does not employ homogeneous transformations, but rather separates out the rotation matrices and positions to achieve computation efficiency. Table 2.1 provides the detailed formulas, with the product of

spatial transforms particularly relevant to the forward kinematics problem.

Kinematic trees are the general structure of robot mechanisms that do not contain closed loops, and the forward kinematics of tree structures are addressed in Chap. 2. The forward kinematics problem for closed chains is much more complicated because of the additional constraints present. Solution methods for closed chains are included in Chap. 12.

1.7 Inverse Kinematics

The inverse kinematics problem for a serial-chain manipulator is to find the values of the joint positions given the position and orientation of the end-effector relative to the base and the values of all of the geometric link parameters. Once again, this is a simplified statement applying only to serial chains. A more general statement is: given the relative positions and orientations of two members of a mechanism, find the values of all of the joint positions. This amounts to finding all of the joint positions given the homogeneous transformation between the two members of interest.

In the common case of a six-degree-of-freedom serial chain manipulator, the known transformation is 0T_6 . Reviewing the formulation of this transformation in Sect. 1.6, it is clear that the inverse kinematics problem for serial-chain manipulators requires the solution of nonlinear sets of equations. In the case of a six-degree-of-freedom manipulator, three of these equations relate to the position vector within the homogeneous transformation, and the other three relate to the rotation matrix. In the latter case, these three equations cannot come from the same row or column because of the dependency within the rotation matrix. With these nonlinear equations, it is possible that no solutions exist or multiple solutions exist [1.33]. For a solution to exist, the desired position and orientation of the end-effector must lie in the workspace of the manipulator. In cases where solutions do exist, they often cannot be presented in closed form, so numerical methods are required.

1.7.1 Closed-Form Solutions

Closed-form solutions are desirable because they are faster than numerical solutions and readily identify all possible solutions. The disadvantage of closed-form solutions is that they are not general, but robot dependent. The most effective methods for finding closed-form so-

lutions are ad hoc techniques that take advantage of particular geometric features of specific mechanisms. In general, closed-form solutions can only be obtained for six-degree-of-freedom systems with special kinematic structure characterized by a large number of the geometric parameters defined in Sect. 1.4 being zero-valued. Most industrial manipulators have such structure because it permits more efficient coordination software. Sufficient conditions for a six-degree-of-freedom manipulator to have closed-form inverse kinematics solutions are [1.34–36]:

1. three consecutive revolute joint axes intersect at a common point, as in a spherical wrist;
2. three consecutive revolute joint axes are parallel.

Closed-form solution approaches are generally divided into algebraic and geometric methods.

Algebraic Methods

Algebraic methods involve identifying the significant equations containing the joint variables and manipulating them into a soluble form. A common strategy is reduction to a transcendental equation in a single variable such as,

$$C_1 \cos \theta_i + C_2 \sin \theta_i + C_3 = 0, \quad (1.45)$$

where C_1 , C_2 , and C_3 are constants. The solution to such an equation is

$$\theta_i = 2 \tan^{-1} \left(\frac{C_2 \pm \sqrt{C_2^2 - C_3^2 + C_1^2}}{C_1 - C_3} \right). \quad (1.46)$$

Special cases in which one or more of the constants are zero are also common.

Reduction to a pair of equations having the form,

$$\begin{aligned} C_1 \cos \theta_i + C_2 \sin \theta_i + C_3 &= 0, \\ C_1 \sin \theta_i - C_2 \cos \theta_i + C_4 &= 0, \end{aligned} \quad (1.47)$$

is another particularly useful strategy because only one solution results,

$$\theta_i = \text{Atan2}(-C_1 C_4 - C_2 C_3, C_2 C_4 - C_1 C_3). \quad (1.48)$$

Geometric Methods

Geometric methods involve identifying points on the manipulator relative to which position and/or orientation can be expressed as a function of a reduced set of the joint variables. This often amounts to decomposing the spatial problem into separate planar problems. The resulting equations are solved using algebraic manipulation. The two sufficient conditions for existence of a closed-form solution for a six-degree-of-freedom manipulator that are listed above enable the decomposition of the problem into inverse position kinematics and inverse orientation kinematics. This is the decomposition into regional and orientation structures discussed in Sect. 1.5, and the solution is found by rewriting (1.44),

$${}^0T_6 {}^6T_5 {}^5T_4 {}^4T_3 = {}^0T_1 {}^1T_2 {}^2T_3. \quad (1.49)$$

The example manipulator in Fig. 1.3 has this structure, and this regional structure is commonly known as an articulated or anthropomorphic arm or an elbow manipulator. The solution to the inverse position kinematics problem for such a structure is summarized in Table 1.9. Because there are two solutions for θ_1 and likewise two solutions for both θ_2 and θ_3 corresponding to each θ_1 solution, there are a total of four solutions to the inverse position kinematics problem of the articulated arm manipulator. The orientation structure is simply a spherical wrist, and the corresponding solution to the inverse orientation kinematics problem is summarized in Table 1.10. Two solutions for θ_5 are given in Table 1.10, but only one solution for both θ_4 and θ_6 corresponds to each. Thus, the inverse orientation kinematics problem of a spherical wrist has two solutions. Combining the regional

Table 1.9 Inverse position kinematics of the articulated arm within the example serial chain manipulator in Fig. 1.3

$$\begin{aligned} \theta_1 &= \text{Atan2}({}^0p_6^y, {}^0p_6^x) \\ &\text{or } \text{Atan2}(-{}^0p_6^y, -{}^0p_6^x), \\ \theta_3 &= -\text{Atan2}(D, \pm\sqrt{1-D^2}), \\ \text{where } D &:= \frac{({}^0p_6^x)^2 + ({}^0p_6^y)^2 + ({}^0p_6^z)^2 - a_3^2 - d_4^2}{2a_3d_4}, \\ \theta_2 &= \text{Atan2}({}^0p_6^z, \sqrt{({}^0p_6^x)^2 + ({}^0p_6^y)^2}) \\ &\quad - \text{Atan2}(d_4 \sin \theta_3, a_3 + d_4 \cos \theta_3) \end{aligned}$$

Table 1.10 Inverse orientation kinematics of the spherical wrist within the example serial chain manipulator in Fig. 1.3, with abbreviations $c_{\theta_i} := \cos \theta_i$ and $s_{\theta_i} := \sin \theta_i$

$$\begin{aligned} \theta_5 &= \text{Atan2}(\pm\sqrt{1-(r_{13}s_{\theta_1}-r_{23}c_{\theta_1})^2}, r_{13}s_{\theta_1}-r_{23}c_{\theta_1}), \\ \theta_4 &= \text{Atan2}(\mp(r_{13}c_{\theta_1}+r_{23}s_{\theta_1})s_{(\theta_2+\theta_3)} \mp r_{33}c_{(\theta_2+\theta_3)}, \\ &\quad \pm(r_{13}s_{\theta_1}+r_{23}c_{\theta_1})c_{(\theta_2+\theta_3)} \mp r_{33}s_{(\theta_2+\theta_3)}), \\ \theta_6 &= \text{Atan2}(\pm(r_{12}s_{\theta_1}+r_{22}c_{\theta_1}), \pm(r_{11}s_{\theta_1}-r_{21}c_{\theta_1})), \\ &\text{where the } \pm \text{ choice for } \theta_5 \text{ dictates all of the subsequent } \pm \\ &\text{and } \mp \text{ for } \theta_4 \text{ and } \theta_6. \end{aligned}$$

and orientation structures, the total number of inverse kinematics solutions for the manipulator in Fig. 1.3 is eight.

1.7.2 Numerical Methods

Unlike the algebraic and geometric methods used to find closed-form solutions, numerical methods are not robot dependent, so they can be applied to any kinematic structure. The disadvantages of numerical methods are that they can be slower and in some cases, they do not allow computation of all possible solutions. For a six-degree-of-freedom serial-chain manipulator with only revolute and prismatic joints, the translation and rotation equations can always be reduced to a polynomial in a single variable of degree not greater than 16 [1.37]. Thus, such a manipulator can have as many as 16 real solutions to the inverse kinematics problem [1.38]. Since closed-form solution of a polynomial equation is only possible if the polynomial is of degree four or less, it follows that many manipulator geometries are not solvable in closed form. In general, a greater number of nonzero geometric parameters corresponds to a polynomial of higher degree in the reduction. For such manipulator structures, the most common numerical methods can be divided into categories of symbolic elimination methods, continuation methods, and iterative methods.

Symbolic Elimination Methods

Symbolic elimination methods involve analytical manipulations to eliminate variables from the system of nonlinear equations to reduce it to a smaller set of equations. *Raghavan* and *Roth* [1.39] used dialytic elimination to reduce the inverse kinematics problem of a general six-revolute serial-chain manipulator to a polynomial of degree 16 and to find all possible solutions. The roots provide solutions for one of the joint variables, while the other variables are computed by solving lin-

ear systems. *Manocha* and *Canny* [1.40] improved the numerical properties of this technique by reformulating the problem as a generalized eigenvalue problem. An alternative approach to elimination makes use of Gröbner bases [1.41, 42].

Continuation Methods

Continuation methods involve tracking a solution path from a start system with known solutions to a target system whose solutions are sought as the start system is transformed into the target system. These techniques have been applied to inverse kinematics problems [1.43], and special properties of polynomial systems can be exploited to find all possible solutions [1.44].

Iterative Methods

A number of different iterative methods can be employed to solve the inverse kinematics problem. Most of them converge to a single solution based on an initial guess, so the quality of that guess greatly impacts

the solution time. Newton–Raphson methods provide a fundamental approach that uses a first-order approximation to the original equations. *Pieper* [1.34] was among the first to apply the method to inverse kinematics, and others have followed [1.45, 46]. Optimization approaches formulate the problem as a nonlinear optimization problem and employ search techniques to move from an initial guess to a solution [1.47, 48]. Resolved motion rate control converts the problem to a differential equation [1.49], and a modified predictor–corrector algorithm can be used to perform the joint velocity integration [1.50]. Control-theory-based methods cast the differential equation into a control problem [1.51]. Interval analysis [1.52] is perhaps one of the most promising iterative methods because it offers rapid convergence to a solution and can be used to find all possible solutions. For complex mechanisms, the damped least-squares approach [1.53] is particularly attractive, and more detail is provided in Chap. 11.

1.8 Forward Instantaneous Kinematics

The forward instantaneous kinematics problem for a serial-chain manipulator is: given the positions of all members of the chain and the rates of motion about all the joints, find the total velocity of the end-effector. Here the rate of motion about the joint is the angular velocity of rotation about a revolute joint or the translational velocity of sliding along a prismatic joint. The total velocity of a member is the velocity of the origin of the reference frame fixed to it combined with its angular velocity. That is, the total velocity has six independent components and therefore, completely represents the velocity field of the member. It is important to note that this definition includes an assumption that the pose of the mechanism is completely known. In most situations, this means that either the forward or inverse position kinematics problem must be solved before the forward instantaneous kinematics problem can be addressed. The same is true of the inverse instantaneous kinematics problem discussed in the following section. The forward instantaneous kinematics problem is important when doing acceleration analysis for the purpose of studying dynamics. The total velocities of the members are needed for the computation of Coriolis and centripetal acceleration components.

1.8.1 Jacobian

Differentiation with respect to time of the forward position kinematics equations yields a set of equations of the form,

$$\mathbf{v}_N = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}, \quad (1.50)$$

where \mathbf{v}_N is the spatial velocity of the end-effector, $\dot{\mathbf{q}}$ is an N -dimensional vector composed of the joint rates, and $\mathbf{J}(\mathbf{q})$ is a $6 \times N$ matrix whose elements are, in gen-

Table 1.11 Algorithm for computing the columns of the Jacobian from the free modes of the joints

n_i	Number of degrees of freedom of joint i
\mathbf{J}	${}^k\mathbf{v}_N = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}$, where k is any frame
\mathbf{J}_{n_i}	n_i column(s) of \mathbf{J} associated with $\dot{\mathbf{q}}_i$,
Φ_ω	first three rows of Φ ,
Φ_v	last three rows of Φ ,
	$\mathbf{J}_{n_i} = {}^k\mathbf{X}_i\Phi_i$,
Expression	Computed value
$\mathbf{X}_1\mathbf{X}_2$	$(\mathbf{R}_1\mathbf{R}_2; \mathbf{p}_2 + \mathbf{R}_2^T\mathbf{p}_1)$
$\mathbf{X}\Phi$	$(\mathbf{R}\Phi_\omega; \mathbf{R}(\Phi_v - \mathbf{p} \times \Phi_\omega))$
\mathbf{X}^{-1}	$(\mathbf{R}^T; -\mathbf{R}\mathbf{p})$
$\mathbf{X}^{-1}\Phi$	$(\mathbf{R}^T\Phi_\omega; \mathbf{R}^T\Phi_v + \mathbf{p} \times \mathbf{R}^T\Phi_\omega)$

eral, nonlinear functions of q_1, \dots, q_N . $J(q)$ is called the Jacobian matrix of this algebraic system and is expressed relative to the same reference frame as the spatial velocity v_N [1.54]. If the joint positions are known, (1.50) yields six linear algebraic equations in the joint rates. If the joint rates are given, a solution of (1.50) is a solution of the forward instantaneous kinematics problem. Note that $J(q)$ can be regarded as a known matrix for this purpose provided all the joint positions are known.

Using the spatial vector notation briefly introduced in Sect. 1.2.6 and explained in detail in Sect. 2.2, the

Jacobian can be easily computed from the free modes Φ_i of the joints and the associated spatial transforms jX_i . The column(s) of $J(q)$ associated with joint rate(s) \dot{q}_i is (are)

$${}^kX_i\Phi_i,$$

where k denotes any reference frame relative to which v_N is expressed. Table 1.11 contains an algorithm for efficiently computing the columns of the Jacobian in this manner. Additional information about Jacobians can be found in Chap. 11.

1.9 Inverse Instantaneous Kinematics

The important problem from the point of view of robotic coordination is the inverse instantaneous kinematics problem. More information on robot coordination can be found in Chaps. 5 and 6. The inverse instantaneous kinematics problem for a serial chain manipulator is: given the positions of all members of the chain and the total velocity of the end-effector, find the rates of motion of all joints. When controlling a movement of an industrial robot which operates in the point-to-point mode, it is not only necessary to compute the final joint positions needed to assume the desired final hand position. It is also necessary to generate a smooth trajectory for motion between the initial and final positions. There are, of course, an infinite number of possible trajectories for this purpose. However, the most straightforward and successful approach employs algorithms based on the solution of the inverse instantaneous kinematics problem. This technique originated in the work of Whitney [1.55] and of Pieper [1.34].

1.9.1 Inverse Jacobian

In order to solve the linear system of equations in the joint rates obtained by decomposing (1.50) into its com-

ponent equations when v_N is known, it is necessary to invert the Jacobian matrix. The equation becomes

$$\dot{q} = J^{-1}(q)v_N. \quad (1.51)$$

Since J is a 6×6 matrix, numerical inversion is not very attractive in real-time software which must run at computation cycle rates on the order of 100 Hz or more. Worse, it is quite possible for J to become singular ($|J| = 0$). The inverse does not then exist. More information on singularities can be found in Chaps. 3 and 12. Even when the Jacobian matrix does not become singular, it may become ill-conditioned, leading to degraded performance in significant portions of the manipulator's workspace. Most industrial robot geometries are simple enough that the Jacobian matrix can be inverted analytically, leading to a set of explicit equations for the joint rates [1.56–58]. This greatly reduces the number of computation operations needed as compared to numerical inversion. For more complex manipulator geometries, though, numerical inversion is the only solution option. The Jacobian of a redundant manipulator is not square, so it cannot be inverted. Chapter 11 discusses how various pseudoinverses can be used in such cases.

1.10 Static Wrench Transmission

Static wrench analysis of a manipulator establishes the relationship between wrenches applied to the end-effector and forces/torques applied to the joints. This is essential for controlling a manipulator's interactions with its environment. Examples include tasks involving fixed or quasi-fixed workpieces such as inserting a component in place with a specified force and tightening a nut to a prescribed torque. More information can be

found in Chaps. 7 and 27. Through the principle of virtual work, the relationship between wrenches applied to the end-effector and forces/torques applied to the joints can be shown to be

$$\tau = J^T f, \quad (1.52)$$

where τ is the n -dimensional vector of applied joint forces/torques for an n -degree-of-freedom manipulator

and \mathbf{f} is the spatial force vector

$$\mathbf{f} = \begin{pmatrix} \mathbf{n} \\ \mathbf{f} \end{pmatrix} \quad (1.53)$$

in which \mathbf{n} and \mathbf{f} are the vectors of torques and forces, respectively, applied to the end-effector, both expressed in the reference frame relative to which

the Jacobian is also expressed. Thus, in the same way the Jacobian maps the joint rates to the spatial velocity of the end-effector, its transpose maps the wrenches applied to the end-effector to the equivalent joint forces/torques. As in the velocity case, when the Jacobian is not square, the inverse relationship is not uniquely defined.

1.11 Conclusions and Further Reading

This chapter presents an overview of how the fundamentals of kinematics can be applied to robotic mechanisms. The topics include various representations of the position and orientation of a rigid body in space, the freedom of motion and accompanying mathematical models of joints, a geometric representation that describes the bodies and joints of a robotics mechanism, the workspace of a manipulator, the problems of forward and inverse kinematics, the problems of forward and inverse instantaneous kinematics including the definition of the Jacobian, and finally the transmission of static wrenches. This chapter is certainly not a comprehensive account of robot kinematics. Fortunately, a number of excellent texts provide a broad introduction to robotics with significant focus on kinematics [1.17,27,29,30,51,59–63].

From a historical perspective, robotics fundamentally changed the nature of the field of mechanism kinematics. Before the first work on the generation of coordination equations for robots [1.34,55], the focus of the field was almost entirely on single-degree-of-freedom mechanisms. This is why robotics, following on from the advent of digital computing, led to a renaissance of work in mechanism kinematics. More details can be found in Chap. 3. The evolution of the field has continued as it has broadened from the study of simple serial chains for industrial robots, the focus of the analysis in this chapter, to parallel machines (see Chap. 12), human-like grippers (Chap. 15), robotic vehicles (Chap. 16 and Chap. 17), and even small-scale robots (see Chap. 18).

References

- 1.1 W. R. Hamilton: On quaternions, or on a new system of imaginaries in algebra, *Philos. Mag.* **18**, installments July 1844 – April 1850, ed. by D. E. Wilkins (2000)
- 1.2 E.B. Wilson: *Vector Analysis* (Dover, New York 1960), based upon the lectures of J. W. Gibbs, (reprint of the second edn. published by Charles Scribner's Sons, 1909)
- 1.3 H. Grassman: *Die Wissenschaft der extensiven Grösse oder die Ausdehnungslehre* (Wigand, Leipzig 1844)
- 1.4 J.M. McCarthy: *Introduction to Theoretical Kinematics* (MIT Press, Cambridge 1990)
- 1.5 W.K. Clifford: Preliminary sketch of bi-quaternions, *Proc. London Math. Soc.*, Vol. 4 (1873) pp. 381–395
- 1.6 A. P. Kotelnikov: Screw calculus and some applications to geometry and mechanics, *Annal. Imp. Univ. Kazan* (1895)
- 1.7 E. Study: *Geometrie der Dynamen* (Teubner, Leipzig 1901)
- 1.8 G.S. Chirikjian, A.B. Kyatkin: *Engineering Applications of Noncommutative Harmonic Analysis* (CRC, Boca Raton 2001)
- 1.9 R. von Mises: Anwendungen der Motorrechnung, *Z. Angew. Math. Mech.* **4**(3), 193–213 (1924)
- 1.10 J.E. Baker, I.A. Parkin: *Fundamentals of Screw Motion: Seminal Papers by Michel Chasles and Olinde Rodrigues* (School of Information Technologies, The University of Sydney, Sydney 2003), translated from O. Rodrigues: Des lois géométriques qui régissent les déplacements d'un système dans l'espace, *J. Math. Pures Applicqu. Liouville* **5**, 380–440 (1840)
- 1.11 R.S. Ball: *A Treatise on the Theory of Screws* (Cambridge Univ Press, Cambridge 1998)
- 1.12 J.K. Davidson, K.H. Hunt: *Robots and Screw Theory: Applications of Kinematics and Statics to Robotics* (Oxford Univ Press, Oxford 2004)
- 1.13 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Clarendon, Oxford 1978)
- 1.14 J.R. Phillips: *Freedom in Machinery: Volume 1. Introducing Screw Theory* (Cambridge Univ Press, Cambridge 1984)
- 1.15 J.R. Phillips: *Freedom in Machinery: Volume 2. Screw Theory Exemplified* (Cambridge Univ Press, Cambridge 1990)

- 1.16 G.S. Chirikjian: Rigid-body kinematics. In: *Robotics and Automation Handbook*, ed. by T. Kurfess (CRC, Boca Raton 2005), Chapt. 2
- 1.17 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)
- 1.18 A. Karger, J. Novak: *Space Kinematics and Lie Groups* (Routledge, New York 1985)
- 1.19 R. von Mises: Motorrechnung, ein neues Hilfsmittel in der Mechanik, *Z. Angew. Math. Mech.* **2**(2), 155–181 (1924), [transl. J. E. Baker, K. Wohlhart, Inst. for Mechanics, T. U. Graz (1996)]
- 1.20 J.D. Everett: On a new method in statics and kinematics, *Mess. Math.* **45**, 36–37 (1875)
- 1.21 R. Featherstone: *Rigid Body Dynamics Algorithms* (Kluwer Academic, Boston 2007)
- 1.22 F. Reuleaux: *Kinematics of Machinery* (Dover, New York 1963), (reprint of *Theoretische Kinematik*, 1875, in German).
- 1.23 K.J. Waldron: A method of studying joint geometry, *Mechan. Machine Theory* **7**, 347–353 (1972)
- 1.24 T.R. Kane, D.A. Levinson: *Dynamics, Theory and Applications* (McGraw-Hill, New York 1985)
- 1.25 J.L. Lagrange: *Oeuvres de Lagrange* (Gauthier-Villars, Paris 1773)
- 1.26 J. Denavit, R.S. Hartenberg: A kinematic notation for lower-pair mechanisms based on matrices, *J. Appl. Mech.* **22**, 215–221 (1955)
- 1.27 W. Khalil, E. Dombre: *Modeling, Identification and Control of Robots* (Taylor Francis, New York 2002)
- 1.28 K.J. Waldron: A study of overconstrained linkage geometry by solution of closure equations, Part I: a method of study, *Mech. Machine Theory* **8**(1), 95–104 (1973)
- 1.29 R. Paul: *Robot Manipulators: Mathematics, Programming and Control* (MIT Press, Cambridge 1982)
- 1.30 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Addison-Wesley, Reading 1986)
- 1.31 K.J. Waldron, A. Kumar: The Dextrous workspace, *ASME Mech. Conf.* (Los Angeles 1980), ASME paper No. 80-DETC-108
- 1.32 R. Vijaykumar, K.J. Waldron, M.J. Tsai: Geometric optimization of manipulator structures for working volume and dexterity, *Int. J. Robot. Res.* **5**(2), 91–103 (1986)
- 1.33 J. Duffy: *Analysis of Mechanisms and Robot Manipulators* (Wiley, New York 1980)
- 1.34 D. Pieper: The Kinematics of Manipulators Under Computer Control. Ph.D. Thesis (Stanford University, Stanford 1968)
- 1.35 C.S.G. Lee: Robot arm kinematics, dynamics, and control, *Computer* **15**(12), 62–80 (1982)
- 1.36 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)
- 1.37 H.Y. Lee, C.G. Liang: A new vector theory for the analysis of spatial mechanisms, *Mechan. Machine Theory* **23**(3), 209–217 (1988)
- 1.38 R. Manseur, K.L. Doty: A robot manipulator with 16 real inverse kinematic solutions, *Int. J. Robot. Res.* **8**(5), 75–79 (1989)
- 1.39 M. Raghavan, B. Roth: Kinematic analysis of the 6R manipulator of general geometry, 5th Int. Symp. Robot. Res. (1990)
- 1.40 D. Manocha, J. Canny: *Real Time Inverse Kinematics for General 6R Manipulators* Tech. rep. (University of California, Berkeley 1992)
- 1.41 B. Buchberger: Applications of Gröbner bases in non-linear computational geometry. In: *Trends in Computer Algebra*, Lect. Notes Comput. Sci., Vol. 296, ed. by R. Janen (Springer, Berlin 1989) pp. 52–80
- 1.42 P. Kovacs: Minimum degree solutions for the inverse kinematics problem by application of the Buchberger algorithm. In: *Advances in Robot Kinematics*, ed. by S. Stifter, J. Lenarcic (Springer, New York 1991) pp. 326–334
- 1.43 L.W. Tsai, A.P. Morgan: Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods, *ASME J. Mechan. Transmission Autom. Design* **107**, 189–195 (1985)
- 1.44 C.W. Wampler, A.P. Morgan, A.J. Sommese: Numerical continuation methods for solving polynomial systems arising in kinematics, *ASME J. Mech. Des.* **112**, 59–68 (1990)
- 1.45 R. Manseur, K.L. Doty: Fast inverse kinematics of 5-revolute-axis robot manipulators, *Mechan. Machine Theory* **27**(5), 587–597 (1992)
- 1.46 S.C.A. Thomopoulos, R.Y.J. Tam: An iterative solution to the inverse kinematics of robotic manipulators, *Mechan. Machine Theory* **26**(4), 359–373 (1991)
- 1.47 J.J. Uicker Jr., J. Denavit, R.S. Hartenberg: An interactive method for the displacement analysis of spatial mechanisms, *J. Appl. Mech.* **31**, 309–314 (1964)
- 1.48 J. Zhao, N. Badler: Inverse kinematics positioning using nonlinear programming for highly articulated figures, *Trans. Comput. Graph.* **13**(4), 313–336 (1994)
- 1.49 D.E. Whitney: Resolved motion rate control of manipulators and human prostheses, *IEEE Trans. Man Mach. Syst.* **10**, 47–63 (1969)
- 1.50 H. Cheng, K. Gupta: A study of robot inverse kinematics based upon the solution of differential equations, *J. Robot. Syst.* **8**(2), 115–175 (1991)
- 1.51 L. Sciavicco, B. Siciliano: *Modeling and Control of Robot Manipulators* (Springer, London 2000)
- 1.52 R.S. Rao, A. Asaithambi, S.K. Agrawal: Inverse Kinematic Solution of Robot Manipulators Using Interval Analysis, *ASME J. Mech. Des.* **120**(1), 147–150 (1998)
- 1.53 C.W. Wampler: Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods, *IEEE Trans. Syst. Man Cybern.* **16**, 93–101 (1986)
- 1.54 D.E. Orin, W.W. Schrader: Efficient computation of the jacobian for robot manipulators, *Int. J. Robot. Res.* **3**(4), 66–75 (1984)

- 1.55 D. E. Whitney: The mathematics of coordinated control of prosthetic arms and manipulators J. Dynamic Sys. Meas. Contr. **122**, 303–309 (1972)
- 1.56 R.P. Paul, B.E. Shimano, G. Mayer: Kinematic control equations for simple manipulators, IEEE Trans. Syst. Man Cybern. **SMC-11**(6), 339–455 (1981)
- 1.57 R.P. Paul, C.N. Stephenson: Kinematics of robot wrists, Int. J. Robot. Res. **20**(1), 31–38 (1983)
- 1.58 R.P. Paul, H. Zhang: Computationally efficient kinematics for manipulators with spherical wrists based on the homogeneous transformation representation, Int. J. Robot. Res. **5**(2), 32–44 (1986)
- 1.59 H. Asada, J.J.E. Slotine: *Robot Analysis and Control* (Wiley, New York 1986)
- 1.60 F.L. Lewis, C.T. Abdallah, D.M. Dawson: *Control of Robot Manipulators* (Macmillan, New York 1993)
- 1.61 R.J. Schilling: *Fundamentals of Robotics: Analysis and Control* (Prentice-Hall, Englewood Cliffs 1990)
- 1.62 M.W. Spong, M. Vidyasagar: *Robot Dynamics and Control* (Wiley, New York 1989)
- 1.63 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)

Dynamics

2. Dynamics

Roy Featherstone, David E. Orin

The dynamic equations of motion provide the relationships between actuation and contact forces acting on robot mechanisms, and the acceleration and motion trajectories that result. Dynamics is important for mechanical design, control, and simulation. A number of algorithms are important in these applications, and include computation of the following: *inverse dynamics*, *forward dynamics*, the *joint-space inertia matrix*, and the *operational-space inertia matrix*. This chapter provides efficient algorithms to perform each of these calculations on a rigid-body model of a robot mechanism. The algorithms are presented in their most general form and are applicable to robot mechanisms with general connectivity, geometry, and joint types. Such mechanisms include fixed-base robots, mobile robots, and parallel robot mechanisms.

In addition to the need for computational efficiency, algorithms should be formulated with a compact set of equations for ease of development and implementation. The use of spatial notation has been very effective in this regard, and is used in presenting the dynamics algorithms. Spatial vector algebra is a concise vector notation for describing rigid-body velocity, acceleration, inertia, etc., using six-dimensional (6-D) vectors and tensors.

The goal of this chapter is to introduce the reader to the subject of robot dynamics and to provide the reader with a rich set of algorithms, in a compact form, that they may apply to their particular robot mechanism. These algorithms are presented in tables for ready access.

2.1	Overview	36
2.1.1	Spatial Vector Notation	36
2.1.2	Canonical Equations	36
2.1.3	Dynamic Models of Rigid-Body Systems	36
2.1.4	Kinematic Trees	37
2.1.5	Kinematic Loops	37
2.2	Spatial Vector Notation	37
2.2.1	Motion and Force	38
2.2.2	Basis Vectors	38
2.2.3	Spatial Velocity and Force	38
2.2.4	Addition and Scalar Multiplication	39
2.2.5	Scalar Product	39
2.2.6	Coordinate Transforms	39
2.2.7	Vector Products	39
2.2.8	Differentiation	40
2.2.9	Acceleration	40
2.2.10	Spatial Momentum	40
2.2.11	Spatial Inertia	41
2.2.12	Equation of Motion	41
2.2.13	Computer Implementation	42
2.2.14	Summary	43
2.3	Canonical Equations	43
2.3.1	Joint-Space Formulation	44
2.3.2	Lagrange Formulation	44
2.3.3	Operational-Space Formulation	44
2.3.4	Impact Model	45
2.4	Dynamic Models of Rigid-Body Systems ..	45
2.4.1	Connectivity	45
2.4.2	Link Geometry	47
2.4.3	Link Inertias	48
2.4.4	Joint Models	48
2.4.5	Example System	48
2.5	Kinematic Trees	50
2.5.1	The Recursive Newton-Euler Algorithm	50
2.5.2	The Articulated-Body Algorithm	53
2.5.3	The Composite-Rigid-Body Algorithm	54
2.5.4	Operational-Space Inertia Matrix ..	55
2.6	Kinematic Loops	57
2.6.1	Formulation of Closed-Loop Algorithm	57
2.6.2	Closed-Loop Algorithm	58
2.7	Conclusions and Further Reading	60
2.7.1	Multibody Dynamics	61
2.7.2	Alternative Representations	61
2.7.3	Alternative Formulations	61

2.7.4 Efficiency	61	2.7.8 Algorithms for Parallel Computers ..	62
2.7.5 Accuracy.....	61	2.7.9 Topologically-Varying Systems	62
2.7.6 Software Packages.....	62		
2.7.7 Symbolic Simplification.....	62	References	62

2.1 Overview

Robot dynamics provides the relationships between actuation and contact forces, and the acceleration and motion trajectories that result. The dynamic equations of motion provide the basis for a number of computational algorithms that are useful in mechanical design, control, and simulation. A growing area of their use is in computer animation of mobile systems, especially using human and humanoid models. In this Chapter, the fundamental dynamic relationships for robot mechanisms are presented, along with efficient algorithms for the most common computations. Spatial vector notation, a concise representation which makes use of 6-D vectors and tensors, is used in the algorithms.

This chapter presents efficient low-order algorithms for four major computations:

1. inverse dynamics, in which the required joint actuator torques/forces are computed from a specification of the robot's trajectory (position, velocity, and acceleration),
2. forward dynamics in which the applied joint actuator torques/forces are specified and the joint accelerations are to be determined,
3. the joint-space inertia matrix, which maps the joint accelerations to the joint torques/forces, and
4. the operational-space inertia matrix, which maps task accelerations to task forces in operational or Cartesian space.

Inverse dynamics is used in feedforward control. Forward dynamics is required for simulation. The joint-space inertia (mass) matrix is used in analysis, in feedback control to linearize the dynamics, and is an integral part of many forward dynamics formulations. The operational-space inertia matrix is used in control at the task or end-effector level.

2.1.1 Spatial Vector Notation

Section 2.2 presents the spatial vector notation, which is used to express the algorithms in this chapter in a clear and concise manner. It was originally developed by *Featherstone* [2.1] to provide a concise vector no-

tation for describing rigid-body velocity, acceleration, inertia, etc., using 6-D vectors and tensors. Section 2.2 explains the meanings of spatial vectors and operators, and provides a detailed tabulation of the correspondence between spatial and standard three-dimensional (3-D) quantities and operators, so that the algorithms in the later sections can be understood. Formulae for efficient computer implementation of spatial arithmetic are also provided. Effort is taken in the discussion of spatial vectors to distinguish between the coordinate vectors and the quantities they represent. This illuminates some of the important characteristics of spatial vectors.

2.1.2 Canonical Equations

The dynamic equations of motion are provided in Sect. 2.3 in two fundamental forms: the joint-space formulation and the operational-space formulation. The terms in the joint-space formulation have traditionally been derived using a Lagrangian approach in which they are developed independently of any reference coordinate frame. The Lagrange formulation provides a description of the relationship between the joint actuator forces and the motion of the mechanism, and fundamentally operates on the kinetic and potential energy in the system. The resulting joint-space formulation has a number of notable properties that have proven useful for developing control algorithms. The equations to relate the terms in the joint-space and operational-space formulations, along with an impact model, are also provided in this section.

2.1.3 Dynamic Models of Rigid-Body Systems

The algorithms in this chapter are model-based and require a data structure describing a robot mechanism as one of their input arguments. Section 2.4 gives a description of the components of this model: a connectivity graph, link geometry parameters, link inertia parameters, and a set of joint models. The description of the connectivity is general so that it covers both kinematic

trees and closed-loop mechanisms. Kinematic trees and the spanning tree for a closed-loop mechanism share a common notation. In order to describe the link and joint geometry, two coordinate frames are associated with each joint, one each attached to the *predecessor* and *successor* links. The successor frame is defined to be compatible with the modified Denavit–Hartenberg convention of Craig [2.2] for single-degree-of-freedom (DOF) joints. The predecessor frame may be defined in a convenient manner to describe a general multi-DOF joint. The relationship between connected links is described using the general joint model of Roberson and Schwertassek [2.3]. A humanoid robot is given as an example to illustrate the link and joint numbering scheme, as well as the assignment of coordinate frames to describe the links and joints. The example includes a floating base, and revolute, universal, and spherical joints.

2.1.4 Kinematic Trees

The algorithms presented in Sect. 2.5 calculate the inverse dynamics, forward dynamics, joint-space inertia matrix, and operational-space inertia matrix for any robot mechanism that is a kinematic tree. An $O(n)$ algorithm for inverse dynamics is provided, where n is the number of degrees of freedom in the mechanism. It uses a Newton–Euler formulation of the problem, and is based on the very efficient recursive Newton–Euler algorithm (RNEA) of Luh, Walker, and Paul [2.4]. Two algorithms are provided for forward dynamics. The first is the $O(n)$ articulated-body algorithm (ABA) which was developed by Featherstone [2.1]. The second is the $O(n^2)$ composite-rigid-body algorithm (CRBA), developed by Walker and Orin [2.5], to compute the joint-space inertia matrix (JSIM). This matrix, together with a vector computed using the RNEA, provide the coefficients of the equation of motion, which can then be solved directly for the accelerations [2.5]. The

operational-space inertia matrix (OSIM) is a kind of articulated-body inertia, and two algorithms are given to calculate it. The first uses the basic definition of the OSIM, and the second is a straightforward $O(n)$ algorithm which is based on efficient solution of the forward dynamics problem. The inputs, outputs, model data, and pseudocode for each algorithm are summarized in tables for ready access.

2.1.5 Kinematic Loops

The above algorithms apply only to mechanisms having the connectivity of kinematic trees, including unbranched kinematic chains. A final algorithm is provided in Sect. 2.6 for the forward dynamics of closed-loop systems, including parallel robot mechanisms. The algorithm makes use of the dynamic equations of motion for a spanning tree of the closed-loop system, and supplements these with loop-closure constraint equations. Three different methods are outlined to solve the resulting linear system of equations. Method 2 is particularly useful if $n \gg n^c$, where n^c is the number of constraints due to the loop-closing joints. This method offers the opportunity to use $O(n)$ algorithms on the spanning tree [2.6]. The section ends with an efficient algorithm to compute the loop-closure constraints by transforming them to a single coordinate system. Since the loop-closure constraint equations are applied at the acceleration level, standard *Baumgarte* stabilization [2.7] is used to prevent the accumulation of position and velocity errors in the loop-closure constraints.

The final section in this chapter provides a conclusion and suggestions for further reading. The area of robot dynamics has been, and continues to be, a very rich area of investigation. This section outlines the major contributions that have been made in the area and the work most often cited. Unfortunately, space does not permit us to provide a comprehensive review of the extensive literature in the area.

2.2 Spatial Vector Notation

There is no single standard notation for robot dynamics. The notations currently in use include 3-D vectors, 4×4 matrices, and several types of 6-D vector: screws, motors, Lie algebra elements, and spatial vectors. Six-dimensional vector notations are generally the best, being more compact than 3-D vectors, and more powerful than 4×4 matrices. We therefore use 6-D vectors throughout this chapter. In particular, we shall use the

spatial vector algebra described in [2.8]. This section provides a brief summary of spatial vectors. Descriptions of 4×4 matrix notations can be found in [2.2, 9], and descriptions of other 6-D vector notations can be found in [2.10–12].

In this handbook, vectors are usually denoted by bold italic letters (e.g., \mathbf{f} , \mathbf{v}). However, to avoid a few name clashes, we shall use upright bold letters to denote spatial

vectors (e.g., \mathbf{f} , \mathbf{v}). Note that this applies only to vectors, not tensors. Also, *in this section only*, we will underline coordinate vectors to distinguish them from the vectors that they represent (e.g., \underline{v} and \underline{v} , representing \mathbf{v} and \mathbf{v}).

2.2.1 Motion and Force

For mathematical reasons, it is useful to distinguish between those vectors that describe the motions of rigid bodies, and those that describe the forces acting upon them. We therefore place motion vectors in a vector space called \mathbf{M}^6 , and force vectors in a space called \mathbf{F}^6 . (The superscripts indicate the dimension.) Motion vectors describe quantities like velocity, acceleration, infinitesimal displacement, and directions of motion freedom; force vectors describe force, momentum, contact normals, and so on.

2.2.2 Basis Vectors

Suppose that \mathbf{v} is a 3-D vector, and that $\underline{v} = (v_x, v_y, v_z)^T$ is the Cartesian coordinate vector that represents \mathbf{v} in the orthonormal basis $\{\hat{x}, \hat{y}, \hat{z}\}$. The relationship between \mathbf{v} and \underline{v} is then given by the formula

$$\mathbf{v} = \hat{x}v_x + \hat{y}v_y + \hat{z}v_z.$$

This same idea applies also to spatial vectors, except that we use Plücker coordinates instead of Cartesian coordinates, and a Plücker basis instead of an orthonormal basis.

Plücker coordinates were introduced in Sect. 1.2.6, but the basis vectors are shown in Fig. 2.1. There are 12 basis vectors in total: six for motion vectors and six for forces. Given a Cartesian coordinate frame, O_{xyz} , the Plücker basis vectors are defined as follows: three unit rotations about the directed lines Ox , Oy , and Oz , denoted by \mathbf{d}_{Ox} , \mathbf{d}_{Oy} , and \mathbf{d}_{Oz} , three unit translations in the directions x , y , and z , denoted by \mathbf{d}_x , \mathbf{d}_y , and \mathbf{d}_z , three unit couples about the x , y , and z directions,

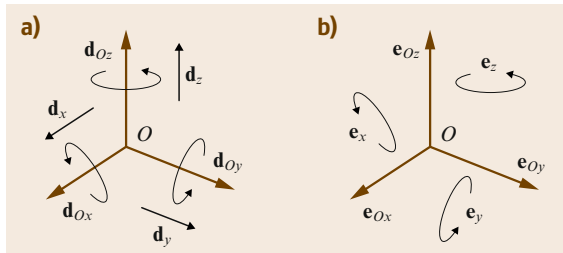


Fig. 2.1a,b Plücker basis vectors for motions (a) and forces (b)

denoted by \mathbf{e}_x , \mathbf{e}_y , and \mathbf{e}_z , and three unit forces along the lines Ox , Oy , and Oz , denoted by \mathbf{e}_{Ox} , \mathbf{e}_{Oy} , and \mathbf{e}_{Oz} .

2.2.3 Spatial Velocity and Force

Given any point O , the velocity of a rigid body can be described by a pair of 3-D vectors, $\boldsymbol{\omega}$ and \mathbf{v}_O , which specify the body's angular velocity and the linear velocity of the body-fixed point currently at O . Note that \mathbf{v}_O is not the velocity of O itself, but the velocity of the body-fixed point that happens to coincide with O at the current instant.

The velocity of this same rigid body can also be described by a single spatial motion vector, $\mathbf{v} \in \mathbf{M}^6$. To obtain \mathbf{v} from $\boldsymbol{\omega}$ and \mathbf{v}_O , we first introduce a Cartesian frame, O_{xyz} , with its origin at O . This frame defines a Cartesian coordinate system for $\boldsymbol{\omega}$ and \mathbf{v}_O , and also a Plücker coordinate system for \mathbf{v} . Given these coordinate systems, it can be shown that

$$\mathbf{v} = \mathbf{d}_{Ox}\omega_x + \mathbf{d}_{Oy}\omega_y + \mathbf{d}_{Oz}\omega_z + \mathbf{d}_x v_{Ox} + \mathbf{d}_y v_{Oy} + \mathbf{d}_z v_{Oz}, \quad (2.1)$$

where ω_x, \dots, v_{Oz} are the Cartesian coordinates of $\boldsymbol{\omega}$ and \mathbf{v}_O in O_{xyz} . Thus, the Plücker coordinates of \mathbf{v} are the Cartesian coordinates of $\boldsymbol{\omega}$ and \mathbf{v}_O . The coordinate vector representing \mathbf{v} in O_{xyz} can be written

$$\underline{v}_O = \begin{pmatrix} \omega_x \\ \vdots \\ v_{Oz} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\omega} \\ \underline{v}_O \end{pmatrix}. \quad (2.2)$$

The notation on the far right of this equation is simply a convenient abbreviation of the list of Plücker coordinates.

The definition of spatial force is very similar. Given any point O , any system of forces acting on a single rigid body is equivalent to a single force \mathbf{f} acting on a line passing through O , together with a pure couple, \mathbf{n}_O , which is the moment of the force system about O . Thus, the two vectors \mathbf{f} and \mathbf{n}_O describe the force acting on a rigid body in much the same way that $\boldsymbol{\omega}$ and \mathbf{v}_O describe its velocity. This same force can also be described by a single spatial force vector, $\mathbf{f} \in \mathbf{F}^6$. Introducing the frame O_{xyz} , as before, it can be shown that

$$\mathbf{f} = \mathbf{e}_x n_{Ox} + \mathbf{e}_y n_{Oy} + \mathbf{e}_z n_{Oz} + \mathbf{e}_{Ox} f_x + \mathbf{e}_{Oy} f_y + \mathbf{e}_{Oz} f_z, \quad (2.3)$$

where n_{Ox}, \dots, f_z are the Cartesian coordinates of \mathbf{n}_O and \mathbf{f} in O_{xyz} . The coordinate vector representing \mathbf{f} in

O_{xyz} can then be written

$$\underline{\mathbf{f}}_O = \begin{pmatrix} n_{Ox} \\ \vdots \\ f_z \end{pmatrix} = \begin{pmatrix} \underline{n}_O \\ \underline{f} \end{pmatrix}. \quad (2.4)$$

Again, these are the Plücker coordinates of \mathbf{f} in O_{xyz} , and the notation on the far right is simply a convenient abbreviation of the list of Plücker coordinates.

2.2.4 Addition and Scalar Multiplication

Spatial vectors behave in the obvious way under addition and scalar multiplication. For example, if \mathbf{f}_1 and \mathbf{f}_2 both act on the same rigid body, then their resultant is $\mathbf{f}_1 + \mathbf{f}_2$; if two different bodies have velocities of \mathbf{v}_1 and \mathbf{v}_2 , then the velocity of the second body relative to the first is $\mathbf{v}_2 - \mathbf{v}_1$; and if \mathbf{f} denotes a force of 1 N acting along a particular line in space, then $\alpha \mathbf{f}$ denotes a force of α N acting along the same line.

2.2.5 Scalar Product

A scalar product is defined between any two spatial vectors, provided that one of them is a motion and the other a force. Given any $\mathbf{m} \in \mathbf{M}^6$ and $\mathbf{f} \in \mathbf{F}^6$, the scalar product can be written either $\mathbf{f} \cdot \mathbf{m}$ or $\mathbf{m} \cdot \mathbf{f}$, and expresses the work done by a force \mathbf{f} acting on a body with motion \mathbf{m} . Expressions like $\mathbf{f} \cdot \mathbf{f}$ and $\mathbf{m} \cdot \mathbf{m}$ are not defined. If $\underline{\mathbf{m}}$ and $\underline{\mathbf{f}}$ are coordinate vectors representing \mathbf{m} and \mathbf{f} in the same coordinate system, then

$$\mathbf{m} \cdot \mathbf{f} = \underline{\mathbf{m}}^T \underline{\mathbf{f}}. \quad (2.5)$$

2.2.6 Coordinate Transforms

Motion and force vectors obey different transformation rules. Let A and B be two coordinate frames, each defining a coordinate system of the same name; and let $\underline{\mathbf{m}}_A$, $\underline{\mathbf{m}}_B$, $\underline{\mathbf{f}}_A$, and $\underline{\mathbf{f}}_B$ be coordinate vectors representing the spatial vectors $\mathbf{m} \in \mathbf{M}^6$ and $\mathbf{f} \in \mathbf{F}^6$ in A and B coordinates, respectively. The transformation rules are then

$$\underline{\mathbf{m}}_B = {}^B\mathbf{X}_A \underline{\mathbf{m}}_A \quad (2.6)$$

and

$$\underline{\mathbf{f}}_B = {}^B\mathbf{X}_A^F \underline{\mathbf{f}}_A, \quad (2.7)$$

where ${}^B\mathbf{X}_A$ and ${}^B\mathbf{X}_A^F$ are the coordinate transformation matrices from A to B for motion and force vectors,

respectively. These matrices are related by the identity

$${}^B\mathbf{X}_A^F \equiv ({}^B\mathbf{X}_A)^{-T} \equiv ({}^A\mathbf{X}_B)^T. \quad (2.8)$$

Suppose that the position and orientation of frame A relative to frame B is described by a position vector ${}^B\mathbf{p}_A$ and a 3×3 rotation matrix ${}^B\mathbf{R}_A$, as described in Sect. 1.2. The formula for ${}^B\mathbf{X}_A$ is then

$$\begin{aligned} {}^B\mathbf{X}_A &= \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ S({}^B\mathbf{p}_A) & \mathbf{1} \end{pmatrix} \begin{pmatrix} {}^B\mathbf{R}_A & \mathbf{0} \\ \mathbf{0} & {}^B\mathbf{R}_A \end{pmatrix} \\ &= \begin{pmatrix} {}^B\mathbf{R}_A & \mathbf{0} \\ S({}^B\mathbf{p}_A){}^B\mathbf{R}_A & {}^B\mathbf{R}_A \end{pmatrix}, \end{aligned} \quad (2.9)$$

and its inverse is

$${}^A\mathbf{X}_B = \begin{pmatrix} {}^A\mathbf{R}_B & \mathbf{0} \\ \mathbf{0} & {}^A\mathbf{R}_B \end{pmatrix} \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ -S({}^B\mathbf{p}_A) & \mathbf{1} \end{pmatrix}. \quad (2.10)$$

The quantity $S(\underline{\mathbf{p}})$ is the skew-symmetric matrix that satisfies $S(\underline{\mathbf{p}})\underline{\mathbf{v}} = \underline{\mathbf{p}} \times \underline{\mathbf{v}}$ for any 3-D vector $\underline{\mathbf{v}}$. It is defined by the equation

$$S(\underline{\mathbf{p}}) = \begin{pmatrix} 0 & -p_z & p_y \\ p_z & 0 & -p_x \\ -p_y & p_x & 0 \end{pmatrix}. \quad (2.11)$$

2.2.7 Vector Products

There are two vector (cross) products defined on spatial vectors. The first takes two motion-vector arguments, and produces a motion-vector result. It is defined by the formula

$$\begin{aligned} \underline{\mathbf{m}}_1 \times \underline{\mathbf{m}}_2 &= \begin{pmatrix} \underline{\mathbf{m}}_1 \\ \underline{\mathbf{m}}_{1O} \end{pmatrix} \times \begin{pmatrix} \underline{\mathbf{m}}_2 \\ \underline{\mathbf{m}}_{2O} \end{pmatrix} \\ &= \begin{pmatrix} \underline{\mathbf{m}}_1 \times \underline{\mathbf{m}}_2 \\ \underline{\mathbf{m}}_1 \times \underline{\mathbf{m}}_{2O} + \underline{\mathbf{m}}_{1O} \times \underline{\mathbf{m}}_2 \end{pmatrix}. \end{aligned} \quad (2.12)$$

The second takes a motion vector as left-hand argument and a force vector as right-hand argument, and produces a force-vector result. It is defined by the formula

$$\begin{aligned} \underline{\mathbf{m}} \times \underline{\mathbf{f}} &= \begin{pmatrix} \underline{\mathbf{m}} \\ \underline{\mathbf{m}}_O \end{pmatrix} \times \begin{pmatrix} \underline{\mathbf{f}} \\ \underline{\mathbf{f}}_O \end{pmatrix} \\ &= \begin{pmatrix} \underline{\mathbf{m}} \times \underline{\mathbf{f}}_O + \underline{\mathbf{m}}_O \times \underline{\mathbf{f}} \\ \underline{\mathbf{m}} \times \underline{\mathbf{f}} \end{pmatrix}. \end{aligned} \quad (2.13)$$

These products arise in differentiation formulae.

It is possible to define a spatial cross-product operator, in analogy with (2.11), as follows:

$$S(\underline{\mathbf{m}}) = \begin{pmatrix} S(\underline{\mathbf{m}}) & \mathbf{0} \\ S(\underline{\mathbf{m}}_O) & S(\underline{\mathbf{m}}) \end{pmatrix}, \quad (2.14)$$

in which case

$$\underline{\mathbf{m}}_1 \times \underline{\mathbf{m}}_2 = S(\underline{\mathbf{m}}_1) \underline{\mathbf{m}}_2, \quad (2.15)$$

but

$$\underline{\mathbf{m}} \times \underline{\mathbf{f}} = -S(\underline{\mathbf{m}})^T \underline{\mathbf{f}}. \quad (2.16)$$

Observe that $S(\underline{\mathbf{m}})$ maps motion vectors to motion vectors, but $S(\underline{\mathbf{m}})^T$ maps force vectors to force vectors.

2.2.8 Differentiation

The derivative of a spatial vector is defined by

$$\frac{d}{dx} \mathbf{s}(x) = \lim_{\delta x \rightarrow 0} \frac{\mathbf{s}(x + \delta x) - \mathbf{s}(x)}{\delta x}, \quad (2.17)$$

where \mathbf{s} here stands for any spatial vector. The derivative is a spatial vector of the same kind (motion or force) as that being differentiated.

The formula for differentiating a spatial coordinate vector in a moving coordinate system is

$$\left(\frac{d}{dt} \mathbf{s} \right)_A = \frac{d}{dt} \underline{\mathbf{s}}_A + \underline{\mathbf{v}}_A \times \underline{\mathbf{s}}_A, \quad (2.18)$$

where \mathbf{s} is any spatial vector, ds/dt is the time derivative of \mathbf{s} , A is the moving coordinate system, $(ds/dt)_A$ is the coordinate vector that represents ds/dt in A coordinates, $\underline{\mathbf{s}}_A$ is the coordinate vector that represents \mathbf{s} in A coordinates, $d\underline{\mathbf{s}}_A/dt$ is the time derivative of $\underline{\mathbf{s}}_A$ (which is the componentwise derivative, since $\underline{\mathbf{s}}_A$ is a coordinate vector), and $\underline{\mathbf{v}}_A$ is the velocity of the A coordinate frame, expressed in A coordinates.

The time derivative of a spatial vector that changes only because it is moving is given by

$$\frac{d}{dt} \mathbf{s} = \mathbf{v} \times \mathbf{s}, \quad (2.19)$$

where \mathbf{v} is the velocity of \mathbf{s} . This formula is useful for differentiating quantities that do not change in their own right, but are attached to moving rigid bodies (e.g., joint axis vectors).

2.2.9 Acceleration

Spatial acceleration is defined as the rate of change of spatial velocity. Unfortunately, this means that spatial

acceleration differs from the classical textbook definition of rigid-body acceleration, which we shall call *classical acceleration*. Essentially, the difference can be summarized as follows:

$$\underline{\mathbf{a}} = \begin{pmatrix} \dot{\underline{\omega}} \\ \dot{\underline{\mathbf{v}}}_O \end{pmatrix} \quad \text{and} \quad \underline{\mathbf{a}}' = \begin{pmatrix} \dot{\underline{\omega}} \\ \dot{\underline{\mathbf{v}}}'_O \end{pmatrix}, \quad (2.20)$$

where $\underline{\mathbf{a}}$ is the spatial acceleration, $\underline{\mathbf{a}}'$ is the classical acceleration, $\underline{\mathbf{v}}_O$ is the derivative of $\underline{\mathbf{v}}_O$ taking O to be fixed in space, and $\underline{\mathbf{v}}'_O$ is the derivative of $\underline{\mathbf{v}}_O$ taking O to be fixed in the body. The two accelerations are related by

$$\underline{\mathbf{a}}' = \underline{\mathbf{a}} + \begin{pmatrix} \mathbf{0} \\ \underline{\omega} \times \underline{\mathbf{v}}_O \end{pmatrix}. \quad (2.21)$$

If \mathbf{r} is a position vector giving the position of the body-fixed point at O relative to any fixed point, then

$$\begin{aligned} \mathbf{v}_O &= \dot{\mathbf{r}}, \\ \dot{\mathbf{v}}_O &= \ddot{\mathbf{r}}, \\ \dot{\mathbf{v}}_O &= \ddot{\mathbf{r}} - \underline{\omega} \times \mathbf{v}_O. \end{aligned} \quad (2.22)$$

The practical difference is that spatial accelerations are easier to use. For example, if the bodies B_1 and B_2 have velocities of \mathbf{v}_1 and \mathbf{v}_2 , respectively, and \mathbf{v}_{rel} is the relative velocity of B_2 with respect to B_1 , then

$$\mathbf{v}_2 = \mathbf{v}_1 + \mathbf{v}_{\text{rel}}.$$

The relationship between their spatial accelerations is obtained simply by differentiating the velocity formula:

$$\frac{d}{dt} (\mathbf{v}_2 = \mathbf{v}_1 + \mathbf{v}_{\text{rel}}) \Rightarrow \underline{\mathbf{a}}_2 = \underline{\mathbf{a}}_1 + \underline{\mathbf{a}}_{\text{rel}}.$$

Observe that spatial accelerations are composed by addition, exactly like velocities. There are no Coriolis or centrifugal terms to worry about. This is a significant improvement on the formulae for composing classical accelerations, such as those in [2.2, 13, 14].

2.2.10 Spatial Momentum

Suppose that a rigid body has a mass of m , a center of mass at C , and a rotational inertia of $\bar{\mathbf{I}}^{\text{cm}}$ about C (Fig. 2.2). If this body is moving with a spatial velocity of $\underline{\mathbf{v}}_C = (\underline{\omega}^T \underline{\mathbf{v}}_C^T)^T$, then its linear momentum is $\mathbf{h} = m\mathbf{v}_C$, and its intrinsic angular momentum is $\mathbf{h}_C = \bar{\mathbf{I}}^{\text{cm}}\underline{\omega}$. Its moment of momentum about a general point, O , is $\mathbf{h}_O = \mathbf{h}_C + \mathbf{c} \times \mathbf{h}$, where $\mathbf{c} = \overrightarrow{OC}$. We can assemble these vectors into a spatial momentum vector as follows:

$$\underline{\mathbf{h}}_C = \begin{pmatrix} \mathbf{h}_C \\ \mathbf{h} \end{pmatrix} = \begin{pmatrix} \bar{\mathbf{I}}^{\text{cm}} \underline{\omega} \\ m \underline{\mathbf{v}}_C \end{pmatrix} \quad (2.23)$$

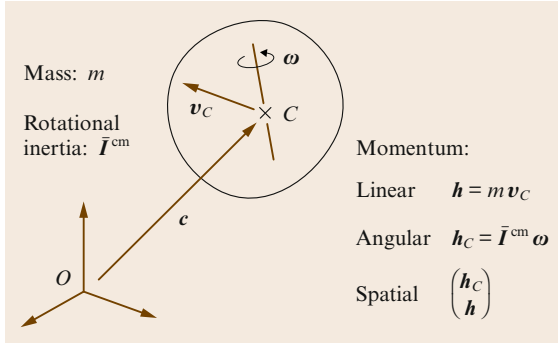


Fig. 2.2 Spatial momentum

and

$$\underline{\mathbf{h}}_O = \begin{pmatrix} \underline{\mathbf{h}}_O \\ \underline{\mathbf{h}} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & S(\underline{\mathbf{c}}) \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \underline{\mathbf{h}}_C. \quad (2.24)$$

Spatial momentum is a force vector, and transforms accordingly.

2.2.11 Spatial Inertia

The spatial momentum of a rigid body is the product of its spatial inertia and velocity:

$$\mathbf{h} = \mathbf{I} \mathbf{v}, \quad (2.25)$$

where \mathbf{I} is the spatial inertia. Expressed in Plücker coordinates at C , we have

$$\mathbf{h}_C = \mathbf{I}_C \mathbf{v}_C, \quad (2.26)$$

which implies

$$\mathbf{I}_C = \begin{pmatrix} \bar{I}^{\text{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix}. \quad (2.27)$$

This is the general formula for the spatial inertia of a rigid body expressed at its center of mass. To express it at another point, O , we proceed as follows. From (2.24), (2.26), and (2.27)

$$\begin{aligned} \underline{\mathbf{h}}_O &= \begin{pmatrix} \mathbf{1} & S(\underline{\mathbf{c}}) \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \bar{I}^{\text{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \underline{\mathbf{v}}_C \\ &= \begin{pmatrix} \mathbf{1} & S(\underline{\mathbf{c}}) \\ \mathbf{0} & \mathbf{1} \end{pmatrix} \begin{pmatrix} \bar{I}^{\text{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \begin{pmatrix} \mathbf{1} & \mathbf{0} \\ S(\underline{\mathbf{c}})^T & \mathbf{1} \end{pmatrix} \underline{\mathbf{v}}_O \\ &= \begin{pmatrix} \bar{I}^{\text{cm}} + mS(\underline{\mathbf{c}})S(\underline{\mathbf{c}})^T & mS(\underline{\mathbf{c}}) \\ mS(\underline{\mathbf{c}})^T & m\mathbf{1} \end{pmatrix} \underline{\mathbf{v}}_O; \end{aligned}$$

but we also have that $\underline{\mathbf{h}}_O = \mathbf{I}_O \underline{\mathbf{v}}_O$, so

$$\mathbf{I}_O = \begin{pmatrix} \bar{I}^{\text{cm}} + mS(\underline{\mathbf{c}})S(\underline{\mathbf{c}})^T & mS(\underline{\mathbf{c}}) \\ mS(\underline{\mathbf{c}})^T & m\mathbf{1} \end{pmatrix}. \quad (2.28)$$

This equation can also be written

$$\mathbf{I}_O = \begin{pmatrix} \bar{I}_O & mS(\underline{\mathbf{c}}) \\ mS(\underline{\mathbf{c}})^T & m\mathbf{1} \end{pmatrix}, \quad (2.29)$$

where

$$\bar{I}_O = \bar{I}^{\text{cm}} + mS(\underline{\mathbf{c}})S(\underline{\mathbf{c}})^T \quad (2.30)$$

is the rotational inertia of the rigid body about O .

Spatial inertia matrices are symmetric and positive-definite. In the general case, 21 numbers are required to specify a spatial inertia (e.g., for an articulated-body or operational-space inertia); but a rigid-body inertia needs only 10 parameters: the mass, the coordinates of the center of mass, and the six independent elements of either \bar{I}^{cm} or \bar{I}_O .

The transformation rule for spatial inertias is

$$\mathbf{I}_B = {}^B\mathbf{X}_A^F \mathbf{I}_A {}^A\mathbf{X}_B, \quad (2.31)$$

where A and B are any two coordinate systems. In practice, we often need to calculate \mathbf{I}_A from \mathbf{I}_B , given only ${}^B\mathbf{X}_A$. The formula for this transformation is

$$\mathbf{I}_A = ({}^B\mathbf{X}_A)^T \mathbf{I}_B {}^B\mathbf{X}_A. \quad (2.32)$$

If two bodies, having inertias \mathbf{I}_1 and \mathbf{I}_2 , are rigidly connected to form a single composite body, then the inertia of the composite, \mathbf{I}_{tot} , is the sum of the inertias of its parts:

$$\mathbf{I}_{\text{tot}} = \mathbf{I}_1 + \mathbf{I}_2. \quad (2.33)$$

This single equation takes the place of three equations in the traditional 3-D vector approach: one to compute the composite mass, one to compute the composite center of mass, and one to compute the composite rotational inertia. If a rigid body with inertia \mathbf{I} is moving with a velocity of \mathbf{v} , then its kinetic energy is

$$T = \frac{1}{2} \mathbf{v} \cdot \mathbf{I} \mathbf{v}. \quad (2.34)$$

If a rigid body, B , is part of a larger system, then it is possible to define an apparent-inertia matrix for B , which describes the relationship between a force acting on B and its resulting acceleration, taking into account the effects of the other bodies in the system. Such quantities are called articulated-body inertias. If B happens to be the end-effector of a robot, then its apparent inertia is called an operational-space inertia.

2.2.12 Equation of Motion

The spatial equation of motion states that the net force acting on a rigid body equals its rate of change of momentum:

$$\mathbf{f} = \frac{d}{dt}(\mathbf{I} \mathbf{v}) = \mathbf{I} \mathbf{a} + \dot{\mathbf{I}} \mathbf{v}.$$

It can be shown that the expression $\dot{\mathbf{I}}\mathbf{v}$ evaluates to $(\mathbf{v} \times \mathbf{I}\mathbf{v})$ [2.8, 15], so the equation of motion can be written

$$\mathbf{f} = \mathbf{I}\mathbf{a} + \mathbf{v} \times \mathbf{I}\mathbf{v}. \quad (2.35)$$

This single equation incorporates both Newton's and Euler's equations of motion for a rigid body. To verify this, we can recover them as follows. Expressing (2.35) at the body's center of mass, and using (2.16), (2.14), and (2.22), we have

$$\begin{aligned} \begin{pmatrix} \underline{n}_C \\ \underline{f} \end{pmatrix} &= \begin{pmatrix} \bar{\mathbf{I}}^{\text{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \begin{pmatrix} \underline{\dot{\omega}} \\ \underline{\dot{v}}_C \end{pmatrix} \\ &\quad - \begin{pmatrix} S(\underline{\omega})^T & S(\underline{v}_C)^T \\ \mathbf{0} & S(\underline{\omega})^T \end{pmatrix} \begin{pmatrix} \bar{\mathbf{I}}^{\text{cm}}\underline{\omega} \\ m\underline{v}_C \end{pmatrix} \\ &= \begin{pmatrix} \bar{\mathbf{I}}^{\text{cm}} & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \begin{pmatrix} \underline{\dot{\omega}} \\ \underline{\ddot{c}} - \underline{\omega} \times \underline{v}_C \end{pmatrix} + \begin{pmatrix} \underline{\omega} \times \bar{\mathbf{I}}^{\text{cm}}\underline{\omega} \\ m\underline{\omega} \times \underline{v}_C \end{pmatrix} \\ &= \begin{pmatrix} \bar{\mathbf{I}}^{\text{cm}}\underline{\dot{\omega}} + \underline{\omega} \times \bar{\mathbf{I}}^{\text{cm}}\underline{\omega} \\ m\underline{\ddot{c}} \end{pmatrix}. \end{aligned} \quad (2.36)$$

2.2.13 Computer Implementation

The easiest way to implement spatial vector arithmetic on a computer is to start with an existing matrix arithmetic tool, like MATLAB® and write (or download from the Web) routines to do the following:

1. calculate $S(\mathbf{m})$ from \mathbf{m} according to (2.14)
2. compose X from R and p according to (2.9)
3. compose I from m , c , and $\bar{\mathbf{I}}^{\text{cm}}$ according to (2.28)

All other spatial arithmetic operations can be performed using standard matrix arithmetic routines. However, some additional routines could usefully be added to this list, such as:

- routines to calculate R from various other representations of rotation
- routines to convert between spatial and 4×4 matrix quantities

This is the recommended approach whenever human productivity is more important than computational efficiency.

If greater efficiency is required, then a more elaborate spatial arithmetic library must be used, in which

1. a dedicated data structure is defined for each kind of spatial quantity, and
2. a suite of calculation routines are provided, each implementing a spatial arithmetic operation by means of an efficient formula.

Some examples of suitable data structures and efficient formulae are shown in Table 2.1. Observe that the suggested data structures for rigid-body inertias and Plücker transforms contain only a third as many numbers as the 6×6 matrices they represent. The efficient arithmetic formulae listed in this table offer cost savings ranging from a factor of 1.5 to a factor of 6 relative to the use of general 6×6 and 6×1 matrix arithmetic. Even more efficient formulae can be found in [2.16].

Table 2.1 Summary of spatial vector notation

Spatial quantities:	
\mathbf{v}	velocity of a rigid body
\mathbf{a}	spatial acceleration of a rigid body ($\mathbf{a} = \dot{\mathbf{v}}$)
\mathbf{a}'	classical description of rigid-body acceleration expressed as a 6-D vector
\mathbf{f}	force acting on a rigid body
\mathbf{I}	inertia of a rigid body
X	Plücker coordinate transform for motion vectors
X^F	Plücker coordinate transform for force vectors ($X^F = X^{-T}$)
${}^B X_A$	Plücker transform from A coordinates to B coordinates
\mathbf{m}	a generic motion vector (any element of M^6)
3-D quantities:	
O	coordinate system origin
\mathbf{r}	position of the body-fixed point at O relative to any fixed point in space
ω	angular velocity of a rigid body
\mathbf{v}_O	linear velocity of the body-fixed point at O ($\mathbf{v}_O = \dot{\mathbf{r}}$)
$\dot{\omega}$	angular acceleration of a rigid body
$\dot{\mathbf{v}}_O$	the derivative of \mathbf{v}_O taking O to be fixed in space
$\dot{\mathbf{v}}'_O$	the derivative of \mathbf{v}_O taking O to be fixed in the body; the classical acceleration of the body-fixed point at O ($\dot{\mathbf{v}}'_O = \ddot{\mathbf{r}}$)
\mathbf{f}	linear force acting on a rigid body, or the resultant of a system of linear forces
\mathbf{n}_O	moment about O of a linear force or system of linear forces
m	mass of a rigid body
\mathbf{c}	position of a rigid body's centre of mass, measured relative to O
\mathbf{h}	first moment of mass of a rigid body, $\mathbf{h} = m\mathbf{c}$; can also denote linear momentum
$\bar{\mathbf{I}}^{\text{cm}}$	moment of inertia about a body's centre of mass
$\bar{\mathbf{I}}$	moment of inertia about O
${}^B R_A$	orthonormal rotation matrix transforming from A coordinates to B coordinates
${}^A p_B$	location of the origin of B coordinates relative to the origin of A coordinates, expressed in A coordinates

Table 2.1 (continued)

Equations:			
$\mathbf{v} = \begin{pmatrix} \omega \\ v_O \end{pmatrix}$	$\mathbf{a} = \begin{pmatrix} \dot{\omega} \\ \dot{v}_O \end{pmatrix} = \begin{pmatrix} \dot{\omega} \\ \ddot{r} - \omega \times \dot{r} \end{pmatrix}$		
$\mathbf{f} = \begin{pmatrix} n_O \\ f \end{pmatrix}$	$\mathbf{a}' = \begin{pmatrix} \dot{\omega} \\ \dot{v}_O \end{pmatrix} = \begin{pmatrix} \dot{\omega} \\ \ddot{r} \end{pmatrix} = \mathbf{a} + \begin{pmatrix} 0 \\ \omega \times v_O \end{pmatrix}$		
$\mathbf{I} = \begin{pmatrix} \bar{\mathbf{I}} & S(h) \\ S(h)^T & m\mathbf{1} \end{pmatrix}$	$\mathbf{I} = \begin{pmatrix} \bar{\mathbf{I}}^{\text{cm}} + mS(c)S(c)^T & mS(c) \\ mS(c)^T & m\mathbf{1} \end{pmatrix}$		
${}^B\mathbf{X}_A = \begin{pmatrix} {}^B\mathbf{R}_A & 0 \\ {}^B\mathbf{R}_A S({}^A\mathbf{p}_B)^T & {}^B\mathbf{R}_A \end{pmatrix}$	$\mathbf{I} = \begin{pmatrix} {}^B\mathbf{R}_A & 0 \\ S({}^B\mathbf{p}_A)^T & {}^B\mathbf{R}_A \end{pmatrix}$		
$\mathbf{v} \cdot \mathbf{f} = \mathbf{f} \cdot \mathbf{v} = \mathbf{v}^T \mathbf{f} = \omega \cdot \mathbf{n}_O + v_O \cdot f$			
$\mathbf{v} \times \mathbf{m} = \begin{pmatrix} \omega \times \mathbf{m} \\ v_O \times \mathbf{m} + \omega \times \mathbf{m}_O \end{pmatrix} = \begin{pmatrix} S(\omega) & 0 \\ S(v_O) & S(\omega) \end{pmatrix} \begin{pmatrix} \mathbf{m} \\ \mathbf{m}_O \end{pmatrix}$			
$\mathbf{v} \times \mathbf{f} = \begin{pmatrix} \omega \times \mathbf{n}_O + v_O \times f \\ \omega \times f \end{pmatrix} = \begin{pmatrix} S(\omega) & S(v_O) \\ 0 & S(\omega) \end{pmatrix} \begin{pmatrix} \mathbf{n}_O \\ f \end{pmatrix}$			
Compact computer representations:			
Mathematical object	Size	Computer representation	Size
$\begin{pmatrix} \omega \\ v_O \end{pmatrix}$	6×1	$(\omega; v_O)$	3+3
$\begin{pmatrix} n_O \\ f \end{pmatrix}$	6×1	$(n_O; f)$	3+3
$\begin{pmatrix} \bar{\mathbf{I}} & S(h) \\ S(h)^T & m\mathbf{1} \end{pmatrix}$	6×6	$(m; h; \bar{\mathbf{I}})$	1+3+9
$\begin{pmatrix} \mathbf{R} & 0 \\ \mathbf{R}S(p)^T & \mathbf{R} \end{pmatrix}$	6×6	$(\mathbf{R}; p)$	9+3

2.2.14 Summary

2.3 Canonical Equations

The equations of motion of a robot mechanism are usually presented in one of two canonical forms: the joint-space formulation,

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}) = \boldsymbol{\tau}, \quad (2.37)$$

or the operational-space formulation,

$$\boldsymbol{\Lambda}(\mathbf{x})\dot{\mathbf{v}} + \boldsymbol{\mu}(\mathbf{x}, \mathbf{v}) + \boldsymbol{\rho}(\mathbf{x}) = \mathbf{f}. \quad (2.38)$$

These equations show the functional dependencies explicitly: \mathbf{H} is a function of \mathbf{q} , $\boldsymbol{\Lambda}$ is a function of \mathbf{x} ,

Table 2.1 (continued)

Efficient spatial arithmetic formulae:	
Expression	Computed value
$\mathbf{X}\mathbf{v}$	$(\mathbf{R}\omega; \mathbf{R}(v_O - p \times \omega))$
$\mathbf{X}^F \mathbf{f}$	$(\mathbf{R}(n_O - p \times f); \mathbf{R}f)$
\mathbf{X}^{-1}	$(\mathbf{R}^T; -\mathbf{R}p)$
$\mathbf{X}^{-1}\mathbf{v}$	$(\mathbf{R}^T\omega; \mathbf{R}^T v_O + p \times \mathbf{R}^T\omega)$
$(\mathbf{X}^F)^{-1}\mathbf{f}$	$(\mathbf{R}^T n_O + p \times \mathbf{R}^T f; \mathbf{R}^T f)$
$\mathbf{X}_1 \mathbf{X}_2$	$(\mathbf{R}_1 \mathbf{R}_2; p_2 + \mathbf{R}_2^T p_1)$
$\mathbf{I}_1 + \mathbf{I}_2$	$(m_1 + m_2; h_1 + h_2; \bar{\mathbf{I}}_1 + \bar{\mathbf{I}}_2)$
$\mathbf{I}\mathbf{v}$	$(\bar{\mathbf{I}}\omega + h \times v_O; m v_O - h \times \omega)$
$\mathbf{X}^T \mathbf{I} \mathbf{X}$	$(m; \mathbf{R}^T h + m p; \mathbf{R}^T \bar{\mathbf{I}} \mathbf{R} - S(p)S(\mathbf{R}^T h) - S(\mathbf{R}^T h + m p)S(p))$
For meaning of $\mathbf{X}^T \mathbf{I} \mathbf{X}$ see (2.32)	

Spatial vectors are 6-D vectors that combine the linear and angular aspects of rigid-body motion, resulting in a compact notation that is very suitable for describing dynamics algorithms. To avoid a few name clashes with 3-D vectors, we have used bold upright letters to denote spatial vectors, while tensors are still denoted by italics. In the sections that follow, upright letters will be used to denote both spatial vectors and vectors that are concatenations of other vectors, like $\dot{\mathbf{q}}$.

Table 2.1 presents a summary of the spatial quantities and operators introduced in this section, together with the formulae that define them in terms of 3-D quantities and operators. It also presents data structures and formulae for efficient computer implementation of spatial arithmetic. This table should be read in conjunction with Tables 1.5 and 1.6, which shows how to compute the orientation, position, and spatial velocities for a variety of joint types. Note that ${}^j\mathbf{R}_i$ and ${}^j\mathbf{p}_i$ in these tables correspond to ${}^B\mathbf{R}_A^T$ and ${}^A\mathbf{p}_B$, respectively, when read in conjunction with Table 2.1.

and so on. Once these dependencies are understood, they are usually omitted. In (2.38), \mathbf{x} is a vector of operational-space coordinates, while \mathbf{v} and \mathbf{f} are spatial vectors denoting the velocity of the end-effector and the external force acting on it. If the robot is redundant, then the coefficients of this equation must be defined as functions of \mathbf{q} and $\dot{\mathbf{q}}$ rather than \mathbf{x} and \mathbf{v} .

These two equations are further explained below, along with a description of the Lagrange formulation of (2.37), and the impulsive equations of motion for impact.

2.3.1 Joint-Space Formulation

The symbols \mathbf{q} , $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, and $\boldsymbol{\tau}$ denote n -dimensional vectors of joint position, velocity, acceleration and force variables, respectively, where n is the number of degrees of motion freedom of the robot mechanism. \mathbf{H} is an $n \times n$ symmetric, positive-definite matrix, and is called the generalized, or joint-space, inertia matrix (JSIM). \mathbf{C} is an $n \times n$ matrix such that $\mathbf{C} \dot{\mathbf{q}}$ is the vector of Coriolis and centrifugal terms (collectively known as *velocity product* terms); and $\boldsymbol{\tau}_g$ is the vector of gravity terms. More terms can be added to this equation, as required, to account for other dynamical effects (e.g., viscous friction). The effects of a force \mathbf{f} exerted on the mechanism at the end-effector can be accounted for by adding the term $\mathbf{J}^T \mathbf{f}$ to the right side of (2.37), where \mathbf{J} is the Jacobian of the end-effector (Sect. 1.8.1).

\mathbf{q} specifies the coordinates of a point in the mechanism's configuration space. If the mechanism is a kinematic tree (see Sect. 2.4), then \mathbf{q} contains every joint variable in the mechanism, otherwise it contains only an independent subset. The elements of \mathbf{q} are generalized coordinates. Likewise, the elements of $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$, and $\boldsymbol{\tau}$ are generalized velocities, accelerations, and forces.

2.3.2 Lagrange Formulation

Various methods exist for deriving the terms in (2.37). The two that are most commonly used in robotics are the Newton–Euler formulation and the Lagrange formulation. The former works directly with Newton's and Euler's equations for a rigid body, which are contained within the spatial equation of motion, (2.35). This formulation is especially amenable to the development of efficient recursive algorithms for dynamics computations, such as those described in Sections 2.5 and 2.6.

The Lagrange formulation proceeds via the Lagrangian of the robot mechanism,

$$L = T - U, \quad (2.39)$$

where T and U are the total kinetic and potential energy, respectively, of the mechanism. The kinetic energy is given by

$$T = \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{H} \dot{\mathbf{q}}. \quad (2.40)$$

The dynamic equations of motion can then be developed using Lagrange's equation for each generalized coordinate:

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i. \quad (2.41)$$

The resulting equation can be written in scalar form:

$$\sum_{j=1}^n H_{ij} \ddot{q}_j + \sum_{j=1}^n \sum_{k=1}^n C_{ijk} \dot{q}_j \dot{q}_k + \tau_{gi} = \tau_i, \quad (2.42)$$

which shows the structure of the velocity-product terms. C_{ijk} are known as Christoffel symbols of the first type, and are given by

$$C_{ijk} = \frac{1}{2} \left(\frac{\partial H_{ij}}{\partial q_k} + \frac{\partial H_{ik}}{\partial q_j} - \frac{\partial H_{jk}}{\partial q_i} \right). \quad (2.43)$$

They are functions of only the position variables, q_i . The elements of \mathbf{C} in (2.37) can be defined as

$$C_{ij} = \sum_{k=1}^n C_{ijk} \dot{q}_k. \quad (2.44)$$

However, \mathbf{C} is not unique, and other definitions are possible.

With the choice of \mathbf{C} given in (2.44), it is possible to show that the matrix \mathbf{N} , given by

$$\mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{\mathbf{H}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}), \quad (2.45)$$

is skew-symmetric [2.17]. Thus, for any $n \times 1$ vector $\boldsymbol{\alpha}$,

$$\boldsymbol{\alpha}^T \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) \boldsymbol{\alpha} = 0. \quad (2.46)$$

This property is quite useful in control, especially when considering $\boldsymbol{\alpha} = \dot{\mathbf{q}}$, which gives

$$\dot{\mathbf{q}}^T \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}} = 0. \quad (2.47)$$

By applying the principle of conservation of energy, it can be shown that (2.47) holds for any choice of the matrix \mathbf{C} [2.17, 18].

2.3.3 Operational-Space Formulation

In (2.38), \mathbf{x} is a 6-D vector of operational-space coordinates giving the position and orientation of the robot's end-effector; \mathbf{v} is the velocity of the end-effector; and \mathbf{f} is the force exerted on the end-effector. \mathbf{x} is typically a list of Cartesian coordinates, and Euler angles or quaternion components, and is related to \mathbf{v} via a differential equation of the form

$$\dot{\mathbf{x}} = \mathbf{E}(\mathbf{x}) \mathbf{v}. \quad (2.48)$$

\mathbf{A} is the operational-space inertia matrix, which is the apparent inertia of the end-effector taking into account the effect of the rest of the robot's mechanism (i.e., it is an articulated-body inertia). $\boldsymbol{\mu}$ and $\boldsymbol{\rho}$ are vectors of velocity-product and gravity terms, respectively.

Operational space (also known as task space) is the space in which high-level motion and force commands are issued and executed. The operational-space formulation is therefore particularly useful in the context of motion and force control systems (Sect. 6.2 and Sect. 7.2). Equation (2.38) can be generalized to operational spaces with dimensions other than six, and to operational spaces that incorporate the motions of more than one end-effector [2.19].

The terms in (2.37) and (2.38) are related by the following formulae

$$\mathbf{v} = \mathbf{J}\dot{\mathbf{q}}, \quad (2.49)$$

$$\dot{\mathbf{v}} = \mathbf{J}\ddot{\mathbf{q}} + \dot{\mathbf{J}}\dot{\mathbf{q}}, \quad (2.50)$$

$$\boldsymbol{\tau} = \mathbf{J}^T \mathbf{f}, \quad (2.51)$$

$$\boldsymbol{\Lambda} = (\mathbf{J}\mathbf{H}^{-1}\mathbf{J}^T)^{-1}, \quad (2.52)$$

$$\boldsymbol{\mu} = \boldsymbol{\Lambda}(\mathbf{J}\mathbf{H}^{-1}\mathbf{C}\dot{\mathbf{q}} - \dot{\mathbf{J}}\dot{\mathbf{q}}), \quad (2.53)$$

and

$$\boldsymbol{\rho} = \boldsymbol{\Lambda}\mathbf{J}\mathbf{H}^{-1}\boldsymbol{\tau}_g. \quad (2.54)$$

These equations assume that $m \leq n$ (m is the dimension of operational-space coordinates), and that the Jacobian \mathbf{J} has full rank. More details can be found in [2.20].

2.3.4 Impact Model

If a robot strikes a rigid body in its environment, then an impulsive force arises at the moment of impact and causes a step change in the robot's velocity. Let us assume that the impact occurs between the end effector and a rigid body in the environment, and that a spatial

impulse of \mathbf{f}' is exerted on the end-effector. This impulse causes a step change of $\Delta\mathbf{v}$ in the end-effector's velocity; and the two are related by the operational-space equation of impulsive motion [2.21],

$$\boldsymbol{\Lambda}\Delta\mathbf{v} = \mathbf{f}'. \quad (2.55)$$

In joint space, the equation of impulsive motion for a robot mechanism is

$$\mathbf{H}\Delta\dot{\mathbf{q}} = \boldsymbol{\tau}', \quad (2.56)$$

where $\boldsymbol{\tau}'$ and $\Delta\dot{\mathbf{q}}$ denote the joint-space impulse and velocity change, respectively. In the case of a collision involving the robot's end-effector, we have

$$\boldsymbol{\tau}' = \mathbf{J}^T \mathbf{f}' \quad (2.57)$$

and

$$\Delta\mathbf{v} = \mathbf{J}\Delta\dot{\mathbf{q}}, \quad (2.58)$$

which follow from (2.51) and (2.49). Equations (2.55)–(2.57) imply that

$$\Delta\dot{\mathbf{q}} = \bar{\mathbf{J}}\Delta\mathbf{v}, \quad (2.59)$$

where $\bar{\mathbf{J}}$ is the inertia-weighted pseudoinverse of \mathbf{J} and is given by

$$\bar{\mathbf{J}} = \mathbf{H}^{-1}\mathbf{J}^T\boldsymbol{\Lambda}. \quad (2.60)$$

$\bar{\mathbf{J}}$ is also known as the dynamically consistent generalized inverse of the Jacobian matrix [2.20]. Note that the expression $\boldsymbol{\Lambda}\mathbf{J}\mathbf{H}^{-1}$, which appears in (2.53) and (2.54), is equal to $\bar{\mathbf{J}}^T$ since \mathbf{H} and $\boldsymbol{\Lambda}$ are both symmetric. Although we have introduced $\bar{\mathbf{J}}$ in the context of impulsive dynamics, it is more typically used in normal (i.e., nonimpulsive) dynamics equations.

2.4 Dynamic Models of Rigid-Body Systems

A basic rigid-body model of a robot mechanism has four components: a connectivity graph, link and joint geometry parameters, link inertia parameters, and a set of joint models. To this model, one can add various force-producing elements, such as springs, dampers, joint friction, actuators, and drives. The actuators and drives, in particular, may have quite elaborate dynamic models of their own. It is also possible to add extra motion freedoms to model elasticity in the joint bearings or links (Chap. 13). This section describes a basic model. More on this topic can be found in books such as [2.3, 8, 22].

2.4.1 Connectivity

A connectivity graph is an undirected graph in which each node represents a rigid body and each arc represents a joint. The graph must be connected; and exactly one node represents a fixed base or reference frame. If the graph represents a mobile robot (i.e., a robot that is not connected to a fixed base), then it is necessary to introduce a fictitious 6-DOF joint between the fixed base and any one body in the mobile robot. The chosen body is then known as a *floating base*. If a single graph is to represent a collection of mobile robots, then each

robot has its own floating base, and each floating base has its own 6-DOF joint. Note that a 6-DOF joint imposes no constraints on the two bodies it connects, so the introduction of a 6-DOF joint alters the connectivity of the graph without altering the physical properties of the system it represents.

In graph-theory terminology, a loop is an arc that connects a node to itself, and a cycle is a closed path that does not traverse any arc more than once. In the connectivity graph of a robot mechanism, loops are not allowed, and cycles are called kinematic loops. A mechanism that contains kinematic loops is called a closed-loop mechanism; and a mechanism that does not is called an open-loop mechanism or a kinematic tree. Every closed-loop mechanism has a spanning tree, which defines an open-loop mechanism, and every joint that is not in the spanning tree is called a loop-closing joint. The joints in the tree are called tree joints.

The fixed base serves as the root node of a kinematic tree, and the root node of any spanning tree on a closed-loop mechanism. A kinematic tree is said to be branched if at least one node has at least two children, and unbranched otherwise. An unbranched kinematic tree is also called a kinematic chain, and a branched tree can be called a branched kinematic chain. A typical industrial robot arm, without a gripper, is a kinematic chain, while a typical humanoid robot is a kinematic tree with a floating base.

In a system containing N_B moving bodies and N_J joints, where N_J includes the 6-DOF joints mentioned above, the bodies and joints are numbered as follows. First, the fixed base is numbered body 0. The other bodies are then numbered from 1 to N_B in any order such that each body has a higher number than its parent. If the system contains kinematic loops then one must first choose a spanning tree, and commit to that choice, since the identity of a body's parent is determined by the spanning tree. This style of numbering is called a *regular numbering scheme*.

Having numbered the bodies, we number the tree joints from 1 to N_B such that joint i connects body i to its parent. The loop-closing joints, if any, are then numbered from $N_B + 1$ to N_J in any order. Each loop-closing joint k closes one independent kinematic loop, and we number the loops from 1 to N_L (where $N_L = N_J - N_B$ is the number of independent loops) such that loop l is the one closed by joint $k = N_B + l$. Kinematic loop l is the unique cycle in the graph that traverses joint k , but does not traverse any other loop-closing joint.

For an unbranched kinematic tree, these rules produce a unique numbering in which the bodies are

numbered consecutively from base to tip, and the joints are numbered such that joint i connects bodies i and $i - 1$. In all other cases, regular numberings are not unique.

Although the connectivity graph is undirected, it is necessary to assign a direction to each joint for the purpose of defining joint velocity and force. This is necessary for both tree joints and loop-closing joints. Specifically, a joint is said to connect from one body to another. We may call them the predecessor $p(i)$ and successor $s(i)$ for joint i , respectively. Joint velocity is then defined as the velocity of the successor relative to the predecessor; and joint force is defined as a force acting on the successor. It is standard practice (but not a necessity) for all tree joints to connect from the parent to the child.

The connectivity of a kinematic tree, or the spanning tree on a closed-loop mechanism, is described by an N_B -element array of parent body numbers, where the i -th element $p(i)$ is the parent of body i . Note that the parent

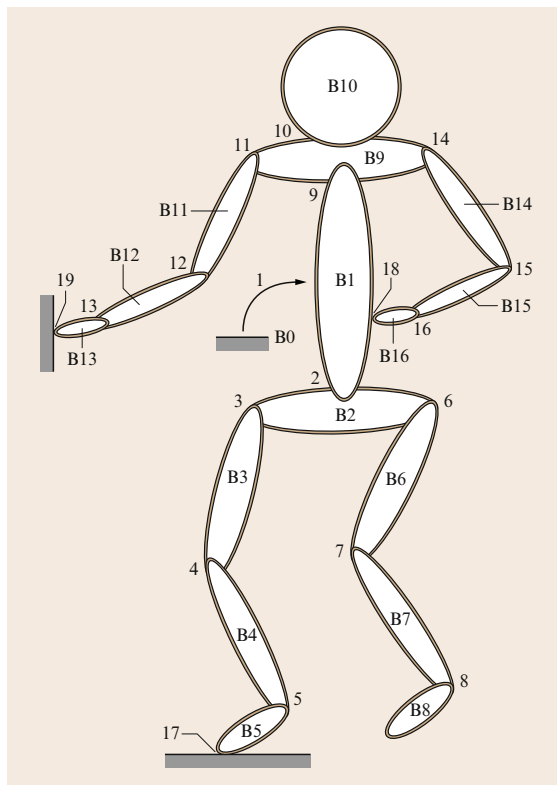


Fig. 2.3 Humanoid robot example. Note: to distinguish between body numbers and joint numbers in this figure, body numbers are preceded by a 'B' for clarity

Table 2.2 Loop-closing joints and roots of loops for the humanoid example

Loop l	Loop-closing joint k	$p(k)$	$s(k)$	Root
1	17	0	5	0
2	18	16	1	1
3	19	0	13	0

$p(i)$ for body i is also the predecessor $p(i)$ for joint i , and thus the common notation. Many algorithms rely on the property $p(i) < i$ to perform their calculations in the correct order. The set of all body numbers for the children of body i , $c(i)$, is also useful in many recursive algorithms.

The connectivity data for kinematic loops may be described in a variety of ways. A representation that facilitates use in recursive algorithms includes the following conventions. Loop-closing joint k joins bodies $p(k)$ (the predecessor) and $s(k)$ (the successor). The set $LR(i)$ for body i gives the numbers of the loops for which body i is the root. Using the property $p(i) < i$ for bodies in the spanning tree, the root of a loop is chosen as the body with the lowest number. In addition, the set $LB(i)$ for body i gives the numbers of the loops to which body i belongs but is not the root.

An example of a closed-loop system is given in Fig. 2.3. The system consists of a humanoid mechanism with topologically varying contacts, with the environment and within the mechanism, which form closed loops. The system has $N_B = 16$ moving bodies and $N_J = 19$ joints with $N_L = N_J - N_B = 3$ loops. The main body (1) is considered to be a *floating base* for this mobile robot system. It is connected to the fixed base (0) through a fictitious 6-DOF joint (1). To complete the example, the loop-closing joint and the body numbers $p(k)$ and $s(k)$ as well as the root body for each loop are given in Table 2.2. The body-based sets $c(i)$ and $LB(i)$ are given in Table 2.3. Note that $LR(0) = \{1, 3\}$ and $LR(1) = \{2\}$ and all other LR sets are null for this example.

2.4.2 Link Geometry

When two bodies are connected by a joint, a complete description of the connection consists of a description of the joint itself, and the locations of two coordinate frames, one in each body, which specify where in each body the joint is located. If there are N_J joints in the system, then there are a total of $2N_J$ joint-attachment frames. One half of these frames are identified with the

Table 2.3 Body-based sets for the humanoid example

Body i	$c(i)$	$LB(i)$	Body i	$c(i)$	$LB(i)$
0	1				
1	2, 9	1, 3	9	10, 11, 14	2, 3
2	3, 6	1	10		
3	4	1	11	12	3
4	5	1	12	13	3
5		1	13		3
6	7		14	15	2
7	8		15	16	2
8			16		2

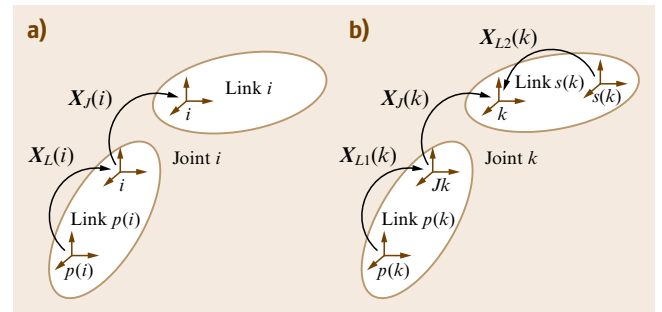
numbers 1 to N_J , and the remainder with the labels $J1$ to JN_J . Each joint i connects from frame Ji to frame i . For joints 1 to N_B (i.e., the tree joints), frame i is rigidly attached to body i . For joints $N_B + 1$ to N_J , frame k for loop-closing joint k will be rigidly attached to body $s(k)$. The second coordinate frame Ji is attached to the predecessor $p(i)$ for each joint i , whether it is a tree joint or a loop-closing joint. Coordinate frame Ji provides a base frame for joint i in that the joint rotation and/or translation is defined relative to this frame.

Figure 2.4 shows the coordinate frames and transforms associated with each joint in the system. The overall transform from frame $p(i)$ coordinates to frame i coordinates for a tree joint is given by:

$${}^iX_{p(i)} = {}^iX_{Ji} {}^{Ji}X_{p(i)} = X_J(i)X_L(i). \quad (2.61)$$

The transform $X_L(i)$ is a fixed link transform which sets the base frame Ji of joint i relative to $p(i)$. It may be used to transform spatial motion vectors from $p(i)$ to Ji coordinates. The transform $X_J(i)$ is a variable joint transform which completes the transformation across joint i from Ji to i coordinates.

Similarly, the overall transform from frame $p(k)$ coordinates to frame k coordinates for a loop-closing joint


Fig. 2.4a,b Coordinate frames and transforms associated with a tree joint (a) and a loop-closing joint (b)

is given by:

$${}^kX_{p(k)} = {}^kX_{Jk} {}^JX_{p(k)} = X_J(k) X_{L1}(k). \quad (2.62)$$

An additional transform $X_{L2}(k)$ is defined from frame $s(k)$ coordinates to frame k coordinates and is given by:

$$X_{L2}(k) = {}^kX_{s(k)}. \quad (2.63)$$

Link and joint geometry data can be specified in a variety of different ways. The most common method is to use Denavit–Hartenberg parameters [2.23]. However, standard Denavit–Hartenberg parameters are not completely general, and are insufficient for describing the geometry for a branched kinematic tree, or for a mechanism containing certain kinds of multi-DOF joints. A modified form of Denavit–Hartenberg parameters [2.2] is used for single-DOF joints in this Handbook (Sect. 1.4). The parameters have been extended for branched kinematic trees [2.22] and closed-loop mechanisms.

2.4.3 Link Inertias

The link inertia data consists of the masses, positions of centers of mass, and rotational inertias of each link in the mechanism. The inertia parameters for link i are expressed in coordinate frame i , and are therefore constants.

2.4.4 Joint Models

The relationship between connected links is described using the general joint model of *Roberson* and *Schwertassek* [2.3]. For a kinematic tree or spanning tree on a closed-loop mechanism, an $n_i \times 1$ vector, $\dot{\mathbf{q}}_i$, relates the velocity of link i to the velocity of its parent, link $p(i)$, where n_i is the number of degrees of freedom at the joint connecting the two links. For a loop-closing joint in a closed-loop mechanism, the relationship is between the velocity of link $s(i)$ (the successor) and the velocity of link $p(i)$ (the predecessor). In either case, the relationship is between the velocity of coordinate frames i and Ji .

Let \mathbf{v}_{rel} and \mathbf{a}_{rel} denote the velocity and acceleration across joint i , that is, the velocity and acceleration of link $s(i)$ relative to $p(i)$. The free modes of the joint are represented by the $6 \times n_i$ matrix Φ_i , such that \mathbf{v}_{rel} and \mathbf{a}_{rel} are given as follows:

$$\mathbf{v}_{\text{rel}} = \Phi_i \dot{\mathbf{q}}_i \quad (2.64)$$

and

$$\mathbf{a}_{\text{rel}} = \Phi_i \ddot{\mathbf{q}}_i + \dot{\Phi}_i \dot{\mathbf{q}}_i, \quad (2.65)$$

where Φ_i and $\dot{\Phi}_i$ depend on the type of joint [2.3]. The matrix Φ_i has full column rank, so we can define a complementary matrix, Φ_i^c , such that the 6×6 matrix $(\Phi_i \Phi_i^c)$ is invertible. We can regard the columns of this matrix as forming a basis on \mathbb{M}^6 such that the first n_i basis vectors define the directions in which motion is allowed, and the remaining $6 - n_i = n_i^c$ vectors define directions in which motion is not allowed. Thus, Φ_i^c represents the constrained modes of joint i .

The force transmitted across joint i from its predecessor to its successor, \mathbf{f}_i , is given as follows:

$$\mathbf{f}_i = (\Psi_i \Psi_i^c) \begin{pmatrix} \boldsymbol{\tau}_i \\ \boldsymbol{\lambda}_i \end{pmatrix}, \quad (2.66)$$

where $\boldsymbol{\tau}_i$ is the $n_i \times 1$ vector of applied forces along the free modes, $\boldsymbol{\lambda}_i$ is the $(6 - n_i) \times 1$ vector of constraint forces, and Ψ_i and Ψ_i^c are computed as follows:

$$(\Psi_i \Psi_i^c) = (\Phi_i \Phi_i^c)^{-T}. \quad (2.67)$$

For most common joint types, it is possible to choose Φ_i and Φ_i^c such that the matrix $(\Phi_i \Phi_i^c)$ is numerically orthonormal, so that $(\Psi_i \Psi_i^c)$ is numerically equal to $(\Phi_i \Phi_i^c)$. Note that (2.67) implies the following relationships: $(\Psi_i)^T \Phi_i = \mathbf{1}_{n_i \times n_i}$, $(\Psi_i)^T \Phi_i^c = \mathbf{0}_{n_i \times (6 - n_i)}$, $(\Psi_i^c)^T \Phi_i = \mathbf{0}_{(6 - n_i) \times n_i}$, and $(\Psi_i^c)^T \Phi_i^c = \mathbf{1}_{(6 - n_i) \times (6 - n_i)}$. When applied to (2.66), the following useful relationship results:

$$\boldsymbol{\tau}_i = \Phi_i^T \mathbf{f}_i. \quad (2.68)$$

The value of $\dot{\Phi}_i$ in (2.65) depends on the type of joint. The general formula is

$$\dot{\Phi}_i = \dot{\Phi}_i + \mathbf{v}_i \times \Phi_i, \quad (2.69)$$

where \mathbf{v}_i is the velocity of link i , and $\dot{\Phi}_i$ is the apparent derivative of Φ_i , as seen by an observer moving with link i , and is given by

$$\dot{\Phi}_i = \frac{\partial \Phi_i}{\partial \mathbf{q}_i} \dot{\mathbf{q}}_i. \quad (2.70)$$

For most common joint types, $\dot{\Phi}_i = \mathbf{0}$.

Single-DOF joints ($n_i = 1$) are especially straightforward to work with when using the Denavit–Hartenberg convention. Motion is chosen along (prismatic) or about (revolute) the $\hat{\mathbf{z}}_i$ coordinate axis. In this case, $\Phi_i = (000001)^T$ for a prismatic joint and $\Phi_i = (001000)^T$ for a revolute joint. Also, $\Phi_i = \mathbf{0}$.

The fictitious 6-DOF joint for a floating base for a mobile robot is also handled relatively easily. For this case, $\Phi_i = \mathbf{1}$ (6×6 identity matrix) and $\dot{\Phi}_i = \mathbf{0}$.

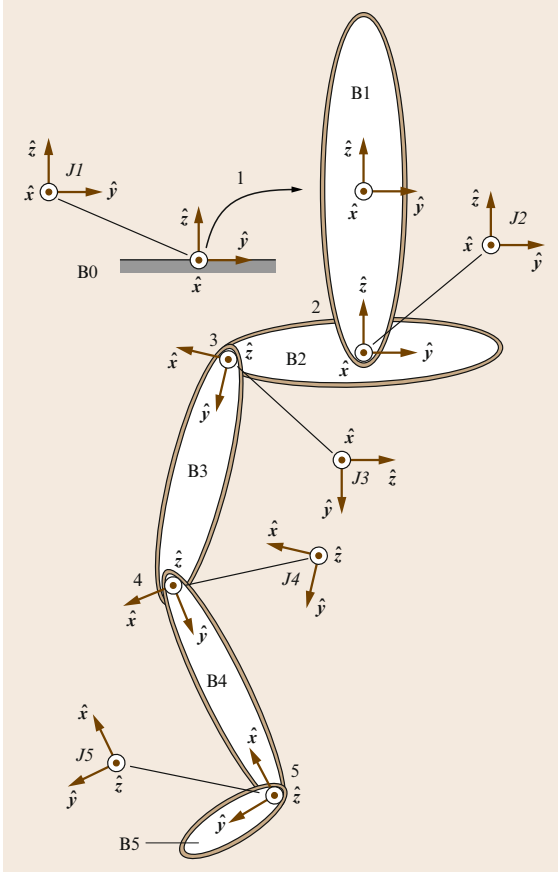


Fig. 2.5 Coordinate frames for the first five links and joints of the humanoid robot example

The revolute joint and floating-base joint, as well as the universal joint ($n_i = 2$) and spherical joint ($n_i = 3$) are illustrated in the example in the next section. For additional details on joint kinematics, see Sect. 1.3.

2.4.5 Example System

In order to illustrate the conventions used for the link and joint models, coordinate frames are attached to the first five links (bodies) and fixed base of the humanoid robot as shown in Fig. 2.5. Note that frame J_i is attached to link $p(i) = i - 1$ for each of the five joints. For this example, the origin of frame J_1 is set coincident with the origin of frame 0, and the origins of frames J_2, J_3, J_4 , and J_5 are coincident with the origins of frames 2, 3, 4, and 5, respectively. Note that J_1 could be set at any position/orientation on the fixed base (B0) to permit the most convenient representation for the motion of the

Table 2.4 Number of degrees of freedom (n_i), fixed rotation (${}^J R_{p(i)}$) and position (${}^{p(i)} p_{Ji}$) from frame $p(i)$ to base frame J_i for joint i of the example system. Note that $2l_i$ is the nominal length of link i along its long axis

Joint	n_i	${}^J R_{p(i)}$	${}^{p(i)} p_{Ji}$
1	6	$\mathbf{1}_{3 \times 3}$	$\mathbf{0}_{3 \times 1}$
2	1	$\mathbf{1}_{3 \times 3}$	$\begin{pmatrix} 0 \\ 0 \\ -l_1 \end{pmatrix}$
3	3	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ -l_2 \\ 0 \end{pmatrix}$
4	1	$\mathbf{1}_{3 \times 3}$	$\begin{pmatrix} 0 \\ 2l_3 \\ 0 \end{pmatrix}$
5	2	$\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 2l_4 \\ 0 \end{pmatrix}$

floating base (B1) relative to the fixed base. Also, the origin of J_2 could be set anywhere along \hat{z}_2 .

The number of degrees of freedom, and fixed rotation and position of the base frames J_i for each joint of the example system, are given in Table 2.4. The rotation ${}^J R_{p(i)}$ transforms 3-D vectors in $p(i)$ coordinates to J_i coordinates. The position ${}^{p(i)} p_{Ji}$ is the vector giving the position of the origin O_{Ji} relative to $O_{p(i)}$, expressed in $p(i)$ coordinates. The spatial transform $\mathbf{X}_{L(i)} = {}^J X_{p(i)}$ may be composed from these 3-D quantities through the equation for ${}^B X_A$ in Table 2.1. The humanoid has a floating base, the torso, a revolute joint between the torso and pelvis (about \hat{z}_2), a spherical joint at the hip, a revolute joint at the knee, and a universal joint at the ankle. As shown in Fig. 2.5, the leg is slightly bent and the foot is turned out to the side ($\approx 90^\circ$ rotation about \hat{y}_3 at the hip).

The free modes, velocity variables, and position variables for all of the joint types in the humanoid are given in Tables 1.5 and 1.6. The expressions for ${}^j R_i$ and ${}^j p_i$ in these tables give ${}^i R_j^T$ and ${}^j p_i$, respectively, through which the joint transform $\mathbf{X}_{J(i)} = {}^i X_{Ji}$ may be composed. Revolute joints follow the Denavit–Hartenberg convention with rotation about the \hat{z}_i axis. The ankle has a pitch rotation of α_5 about the \hat{z}_{J5} axis followed by a roll rotation of β_5 about the \hat{y}_5 axis (see the Z–Y–X Euler angle definitions in Table 1.1). The hip is modeled as a ball-and-socket, spherical joint. To avoid the singularities that are associated with Euler angles, the quaternion ϵ_i may be used

to represent the orientation at the hip. The relationship between the quaternion rate $\dot{\epsilon}_i$ and the relative rotation rate $\omega_{i \text{ rel}}$ is given in (1.7) of the Handbook.

The floating base uses the position of the torso 0p_1 and quaternion ϵ_1 for its position and orientation state variables, respectively. The position of the torso may

be computed by integrating the velocity for the link, as expressed in fixed base coordinates: ${}^0v_1 = {}^0R_1 v_1$, where v_1 is the velocity of the torso in moving coordinates.

Note that $\Phi_i = 0$ for all joints except the universal joint. Since the components of \hat{z}_{J5} in link 5 coordinates vary with $\beta_5, \dot{\beta}_5 \neq 0$. See Sect. 1.3 of the Handbook for further details of the joint kinematics.

2.5 Kinematic Trees

The dynamics of a kinematic tree is simpler, and easier to calculate, than the dynamics of a closed-loop mechanism. Indeed, many algorithms for closed-loop mechanisms work by first calculating the dynamics of a spanning tree, and then subjecting it to the loop-closure constraints.

This section describes the following dynamics algorithms for kinematic trees: the recursive Newton–Euler algorithm (RNEA) for inverse dynamics, the articulated-body algorithm (ABA) for forward dynamics, the composite-rigid-body algorithm (CRBA) for calculating the joint-space inertia matrix (JSIM), and two algorithms to calculate the operational-space inertia matrix (OSIM).

2.5.1 The Recursive Newton–Euler Algorithm

This is an $O(n)$ algorithm for calculating the inverse dynamics of a fixed-base kinematic tree, and is based on the very efficient RNEA of Luh et al. [2.4]. A floating-base version can be found in [2.8, 15]. Given the joint position and velocity variables, this algorithm calculates the applied joint torque/force variables required to produce a given set of joint accelerations.

The link velocities and accelerations are first computed through an outward recursion from the fixed base to the leaf links of the tree. The resultant forces on each link are computed using the Newton–Euler equations (2.35) during this recursion. A second, inward recursion uses the force balance equations at each link to compute the spatial force across each joint and the value of each joint torque/force variable. The key step for computational efficiency is to refer most quantities to local link coordinates. Also, the effects of gravity on each link are efficiently included in the equations by accelerating the base of the mechanism upward.

The calculation proceeds in four steps, as follows, with two steps in each of the two recursions.

Step 1

Calculate the velocity and acceleration of each link in turn, starting with the known velocity and acceleration of the fixed base, and working towards the tips (i. e., the leaf nodes in the connectivity graph).

The velocity of each link in a kinematic tree is given by the recursive formula

$$\mathbf{v}_i = \mathbf{v}_{p(i)} + \Phi_i \dot{\mathbf{q}}_i, \quad (\mathbf{v}_0 = \mathbf{0}), \quad (2.71)$$

where \mathbf{v}_i is the velocity of link i , Φ_i is the motion matrix of joint i , and $\dot{\mathbf{q}}_i$ is the vector of joint velocity variables for joint i .

The equivalent formula for accelerations is obtained by differentiating (2.71), giving

$$\mathbf{a}_i = \mathbf{a}_{p(i)} + \Phi_i \ddot{\mathbf{q}}_i + \dot{\Phi}_i \dot{\mathbf{q}}_i, \quad (\mathbf{a}_0 = \mathbf{0}) \quad (2.72)$$

where \mathbf{a}_i is the acceleration of link i , and $\ddot{\mathbf{q}}_i$ is the vector of joint acceleration variables.

The effect of a uniform gravitational field on the mechanism can be simulated by initializing \mathbf{a}_0 to $-\mathbf{a}_g$ instead of zero, where \mathbf{a}_g is the gravitational acceleration vector. In this case, \mathbf{a}_i is not the true acceleration of link i , but the sum of its true acceleration and $-\mathbf{a}_g$.

Step 2

Calculate the equation of motion for each link. This step computes the forces required to cause the accelerations calculated in step 1. The equation of motion for link i is

$$\mathbf{f}_i^a = \mathbf{I}_i \mathbf{a}_i + \mathbf{v}_i \times \mathbf{I}_i \mathbf{v}_i, \quad (2.73)$$

where \mathbf{I}_i is the spatial inertia of link i , and \mathbf{f}_i^a is the net force acting on link i .

Step 3

Calculate the spatial force across each joint. Referring to Fig. 2.6, the net force acting on link i is

$$\mathbf{f}_i^a = \mathbf{f}_i^e + \mathbf{f}_i - \sum_{j \in c(i)} \mathbf{f}_j,$$

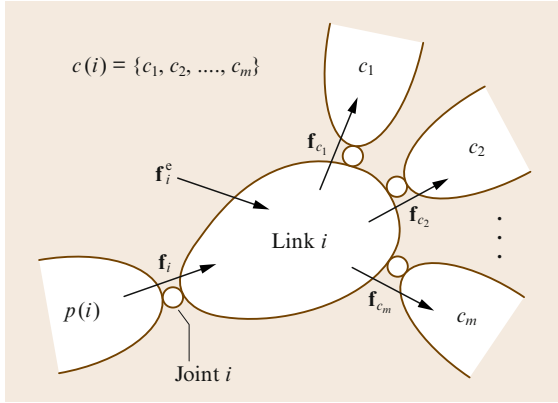


Fig. 2.6 Forces acting on link i

where \mathbf{f}_i is the force transmitted across joint i , \mathbf{f}_i^e is the sum of all relevant external forces acting on link i , and $c(i)$ is the set of children of link i . Rearranging this equation gives the following recursive formula for calculating the joint forces:

$$\mathbf{f}_i = \mathbf{f}_i^a - \mathbf{f}_i^e + \sum_{j \in c(i)} \mathbf{f}_j, \quad (2.74)$$

where i iterates from N_B to 1.

\mathbf{f}_i^e may include contributions from springs, dampers, force fields, contact with the environment, and so on, but its value is assumed to be known, or at least to be calculable from known quantities. If gravity has not been simulated by a fictitious base acceleration, then

Table 2.5 Coordinate-free recursive Newton–Euler algorithm (RNEA) for inverse dynamics

```

v0 = 0
a0 = -ag
for i = 1 to NB do
    vi = vp(i) + Φi q̇i
    ai = ap(i) + Φi q̈i + q̇i
    fi = Ii ai + vi × Ii vi - fie
end
for i = NB to 1 do
    τi = ΦiT fi
    if p(i) ≠ 0 then
        fp(i) = fp(i) + fi
    end
end
end

```

the gravitational force acting on link i must be included in \mathbf{f}_i^e .

Step 4

Calculate the joint force variables, τ_i . By definition, they are given by the equation

$$\tau_i = \Phi_i^T \mathbf{f}_i. \quad (2.75)$$

Coordinate-Free Algorithm

Equations (2.71)–(2.75) imply the algorithm shown in Table 2.5, which is the coordinate-free version of the RNEA. This is the simplest form of the algorithm, and it is suitable for mathematical analysis and related purposes. However, it is not suitable for numerical computation because a numerical version must use coordinate vectors.

Link-Coordinates Algorithm

In general, we say that an algorithm is implemented in link coordinates if a coordinate system is defined for each link, and the calculations pertaining to link i are performed in the coordinate system associated with link i . The alternative is to implement the algorithm in absolute coordinates, in which case all calculations are performed in a single coordinate system, typically that of the base link. In practice, the RNEA is more computationally efficient when implemented in link coordinates, and the same is true of most other dynamics algorithms.

To convert the RNEA to link coordinates, we first examine the equations to see which ones involve quantities from more than one link. Equations (2.73) and (2.75) each involve quantities pertaining to link i only, and therefore need no modification. Such equations are said to be local to link i . The remaining equations involve quantities from more than one link, and therefore require the insertion of coordinate transformation matrices. The modified versions of (2.71), (2.72), and (2.74) are

$$\mathbf{v}_i = {}^i X_{p(i)} \mathbf{v}_{p(i)} + \Phi_i \dot{\mathbf{q}}_i, \quad (2.76)$$

$$\mathbf{a}_i = {}^i X_{p(i)} \mathbf{a}_{p(i)} + \Phi_i \ddot{\mathbf{q}}_i + \dot{\Phi}_i \dot{\mathbf{q}}_i, \quad (2.77)$$

and

$$\mathbf{f}_i = \mathbf{f}_i^a - {}^i X_0^{F0} \mathbf{f}_i^e + \sum_{j \in c(i)} {}^i X_j^F \mathbf{f}_j. \quad (2.78)$$

Equation (2.78) assumes that external forces are expressed in absolute (i. e., link 0) coordinates.

The complete algorithm is shown in Table 2.6. The function `jtype` returns the type code for joint i ; the

Table 2.6 Recursive Newton–Euler algorithm using spatial vectors

```

inputs:  $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, model, {}^0\mathbf{f}_i^e$ 
output:  $\boldsymbol{\tau}$ 
model data:  $N_B, jtype(i), p(i), X_L(i), I_i$ 

 $\mathbf{v}_0 = \mathbf{0}$ 
 $\mathbf{a}_0 = -\mathbf{a}_g$ 
for  $i = 1$  to  $N_B$  do
     $X_J(i) = xjcalc(jtype(i), q_i)$ 
     ${}^iX_{p(i)} = X_J(i) X_L(i)$ 
    if  $p(i) \neq 0$  then
         ${}^iX_0 = {}^iX_{p(i)} {}^{p(i)}X_0$ 
    end
     $\Phi_i = pcalc(jtype(i), q_i)$ 
     $\Phi_{c_i} = pdcalc(jtype(i), q_i, \dot{q}_i)$ 
     $\mathbf{v}_i = {}^iX_{p(i)} \mathbf{v}_{p(i)} + \Phi_i \dot{q}_i$ 
     $\zeta_i = \Phi_{c_i} \dot{q}_i + \mathbf{v}_i \times \Phi_i \dot{q}_i$ 
     $\mathbf{a}_i = {}^iX_{p(i)} \mathbf{a}_{p(i)} + \Phi_i \ddot{q}_i + \zeta_i$ 
     $\mathbf{f}_i = I_i \mathbf{a}_i + \mathbf{v}_i \times I_i \mathbf{v}_i - {}^iX_0^{-T} {}^0\mathbf{f}_i^e$ 
end
for  $i = N_B$  to  $1$  do
     $\boldsymbol{\tau}_i = \Phi_i^T \mathbf{f}_i$ 
    if  $p(i) \neq 0$  then
         $\mathbf{f}_{p(i)} = \mathbf{f}_{p(i)} + {}^iX_{p(i)}^T \mathbf{f}_i$ 
    end
end

```

function `xjcalc` calculates the joint transformation matrix for the specified type of joint; and the functions `pcalc` and `pdcalc` calculate Φ_i and Φ_{c_i} . The formulae used by these functions for a variety of joint types can be found in Tables 1.5 and 1.6. In the general case, both `pcalc` and `pdcalc` are needed. However, for most common joint types, Φ_i is a known constant in link coordinates, and Φ_{c_i} is therefore zero. If it is known in advance that all joints will have this property, then the algorithm can be simplified accordingly. The quantities I_i and $X_L(i)$ are known constants in link coordinates, and are part of the data structure describing the robot mechanism.

The last assignment in the first loop initializes each \mathbf{f}_i to the expression $\mathbf{f}_i^a - {}^iX_0^T {}^0\mathbf{f}_i^e$ (using the identity ${}^iX_0^F = {}^iX_0^{-T}$). The summation on the right-hand side of (2.78) is then performed in the second loop. This al-

Table 2.7 Recursive Newton–Euler algorithm in 3-D vectors, for revolute joints only

```

inputs:  $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, model$ 
output:  $\boldsymbol{\tau}$ 
model data:  $N_B, p(i), R_L(i), {}^{p(i)}\mathbf{p}_i, m_i, c_i, \bar{I}_i^{cm}$ 

 $\boldsymbol{\omega}_0 = \mathbf{0}$ 
 $\dot{\boldsymbol{\omega}}_0 = \mathbf{0}$ 
 $\dot{\mathbf{v}}_0 = -\dot{\mathbf{v}}_g$ 
for  $i = 1$  to  $N_B$  do
     ${}^iR_{p(i)} = \text{rotz}(q_i) R_L(i)$ 
     $\boldsymbol{\omega}_i = {}^iR_{p(i)} \boldsymbol{\omega}_{p(i)} + \hat{\mathbf{z}}_i \dot{q}_i$ 
     $\dot{\boldsymbol{\omega}}_i = {}^iR_{p(i)} \dot{\boldsymbol{\omega}}_{p(i)} + ({}^iR_{p(i)} \boldsymbol{\omega}_{p(i)}) \times \hat{\mathbf{z}}_i \dot{q}_i + \hat{\mathbf{z}}_i \ddot{q}_i$ 
     $\dot{\mathbf{v}}_i = {}^iR_{p(i)} (\dot{\mathbf{v}}_{p(i)} + \dot{\boldsymbol{\omega}}_{p(i)} \times {}^{p(i)}\mathbf{p}_i$ 
         $\quad + \boldsymbol{\omega}_{p(i)} \times \boldsymbol{\omega}_{p(i)} \times {}^{p(i)}\mathbf{p}_i)$ 
     $\mathbf{f}_i = m_i (\dot{\mathbf{v}}_i + \dot{\boldsymbol{\omega}}_i \times c_i + \boldsymbol{\omega}_i \times \boldsymbol{\omega}_i \times c_i)$ 
     $\mathbf{n}_i = \bar{I}_i^{cm} \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \bar{I}_i^{cm} \boldsymbol{\omega}_i + c_i \times \mathbf{f}_i$ 
end
for  $i = N_B$  to  $1$  do
     $\boldsymbol{\tau}_i = \hat{\mathbf{z}}_i^T \mathbf{n}_i$ 
    if  $p(i) \neq 0$  then
         $\mathbf{f}_{p(i)} = \mathbf{f}_{p(i)} + {}^iR_{p(i)}^T \mathbf{f}_i$ 
         $\mathbf{n}_{p(i)} = \mathbf{n}_{p(i)} + {}^iR_{p(i)}^T \mathbf{n}_i + {}^{p(i)}\mathbf{p}_i \times {}^iR_{p(i)}^T \mathbf{f}_i$ 
    end
end

```

gorithm includes code to calculate iX_0 , which is used to transform the external forces to link coordinates. If there are no external forces, then this code can be omitted. If there is only a single external force (e.g., a force at the end-effector of a robot arm) then this code can be replaced with code that transforms the external force vector successively from one link coordinate system to the next, using ${}^iX_{p(i)}$.

Note: although the phrase ‘link coordinates’ suggests that we are using moving coordinate frames, the algorithm is in fact implemented in stationary coordinates that happen to coincide with the moving coordinates at the current instant.

3-D Vector RNEA

The original version of the RNEA was developed and expressed using 3-D vectors (e.g. [2.2,4]). Table 2.7 shows a special case of this algorithm, in which the joints are assumed to be revolute, and the joint axes are assumed

to coincide with the z axes of the link coordinate systems. (Without these assumptions, the equations would be a lot longer.) It also assumes that the external forces are zero.

In this algorithm, $\dot{\mathbf{v}}'_g$ is the linear acceleration due to gravity, expressed in base (link 0) coordinates; `rotz` computes a rotation matrix representing a rotation of the coordinate frame about the z axis; $\mathbf{R}_L(i)$ is the rotational component of $\mathbf{X}_L(i)$; $\mathbf{R}_{p(i)}$ is the rotational component of $\mathbf{X}_{p(i)}$; `pcalc` and `pdcalc` are not used because Φ_i is the known constant $(\hat{\mathbf{z}}^T \mathbf{0}^T)^T$; $\dot{\mathbf{v}}'_i$ is the linear acceleration of the origin of link i coordinates (O_i), and is the linear component of the classical acceleration of link i ; ${}^{p(i)}\mathbf{p}_i$ is the position of O_i relative to $O_{p(i)}$ expressed in $p(i)$ coordinates; and m_i , \mathbf{c}_i , and \mathbf{I}_i^{cm} are the inertia parameters of link i . (See Table 2.1 for the equations relating these 3-D quantities to the corresponding spatial quantities.)

At first sight, the 3-D vector algorithm looks significantly different from the spatial vector algorithm. Nevertheless, it can be obtained directly from the spatial vector algorithm simply by expanding the spatial vectors to their 3-D components, restricting the joint type to revolute, converting spatial accelerations to classical accelerations (i.e., replacing each instance of $\dot{\mathbf{v}}_i$ with $\dot{\mathbf{v}}'_i - \boldsymbol{\omega}_i \times \mathbf{v}_i$ as per (2.22)), and applying some 3-D vector identities to bring the equations into the form shown in the table. The conversion from spatial to classical acceleration has one interesting side-effect: \mathbf{v}_i cancels out of the equation of motion, and therefore does not need to be calculated. As a result, the 3-D version of the algorithm has a slight speed advantage over the spatial version.

2.5.2 The Articulated-Body Algorithm

The **ABA** is an $O(N_B)$ algorithm for calculating the forward dynamics of a kinematic tree. However, under normal circumstances, $O(N_B) = O(n)$, so we shall refer to it as an $O(n)$ algorithm. The **ABA** was developed by Featherstone [2.1] and is an example of a constraint-propagation algorithm. Given the joint position, velocity, and applied torque/force variables, this algorithm calculates the joint accelerations. With the joint accelerations determined, numerical integration may be used to provide a simulation of the mechanism.

The key concept in the **ABA** is illustrated in Fig. 2.7. The subtree rooted at link i interacts with the rest of the kinematic tree only through a force \mathbf{f}_i that is transmitted across joint i . Suppose we break the tree at this point, and consider only the motion of the subtree subject to an

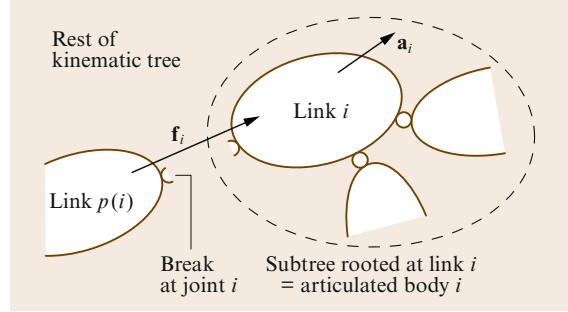


Fig. 2.7 Definition of articulated body i

unknown force, \mathbf{f}_i , acting on link i . It is possible to show that the acceleration of link i is related to the applied force according to the equation

$$\mathbf{f}_i = \mathbf{I}_i^A \mathbf{a}_i + \mathbf{p}_i^A, \quad (2.79)$$

where \mathbf{I}_i^A is called the articulated-body inertia of link i in the subtree (which we can now call an articulated body), and \mathbf{p}_i^A is the associated bias force, which is the force required to produce zero acceleration in link i . Note that \mathbf{p}_i^A depends on the velocities of the individual bodies in the articulated body. Equation (2.79) takes into account the complete dynamics of the subtree. Thus, if we happened to know the correct value of \mathbf{f}_i , then (2.79) would immediately give us the correct acceleration of link i .

The reason we are interested in the quantities \mathbf{I}_i^A and \mathbf{p}_i^A is that they allow us to calculate $\ddot{\mathbf{q}}_i$ from $\mathbf{a}_{p(i)}$, which in turn allows us to calculate \mathbf{a}_i , which then allows us to calculate more joint accelerations, and so on. Combining (2.79) with (2.75) and (2.72) gives

$$\boldsymbol{\tau}_i = \Phi_i^T \mathbf{f}_i = \Phi_i^T (\mathbf{I}_i^A (\mathbf{a}_{p(i)} + \Phi_i \ddot{\mathbf{q}}_i + \dot{\Phi}_i \dot{\mathbf{q}}_i) + \mathbf{p}_i^A),$$

which can be solved for $\ddot{\mathbf{q}}_i$ to give

$$\ddot{\mathbf{q}}_i = \mathbf{D}_i (\mathbf{u}_i - \mathbf{U}_i^T \mathbf{a}_{p(i)}), \quad (2.80)$$

where

$$\begin{aligned} \mathbf{U}_i &= \mathbf{I}_i^A \Phi_i, \\ \mathbf{D}_i &= (\Phi_i^T \mathbf{U}_i)^{-1} = (\Phi_i^T \mathbf{I}_i^A \Phi_i)^{-1}, \\ \mathbf{u}_i &= \boldsymbol{\tau}_i - \mathbf{U}_i^T \zeta_i - \Phi_i^T \mathbf{p}_i^A \end{aligned}$$

and

$$\zeta_i = \dot{\Phi}_i \dot{\mathbf{q}}_i = \ddot{\Phi}_i \dot{\mathbf{q}}_i + \mathbf{v}_i \times \Phi_i \dot{\mathbf{q}}_i.$$

\mathbf{a}_i can then be calculated via (2.72).

It turns out that the articulated-body inertias and bias forces can be calculated efficiently via the recursive formulae

$$\mathbf{I}_i^A = \mathbf{I}_i + \sum_{j \in c(i)} (\mathbf{I}_j^A - \mathbf{U}_j \mathbf{D}_j \mathbf{U}_j^T) \quad (2.81)$$

and

$$\mathbf{p}_i^A = \mathbf{p}_i + \sum_{j \in c(i)} (\mathbf{p}_j^A + \mathbf{I}_j^A \boldsymbol{\zeta}_j + \mathbf{U}_j \mathbf{D}_j \mathbf{u}_j), \quad (2.82)$$

where

$$\mathbf{p}_i = \mathbf{v}_i \times \mathbf{I}_i \mathbf{v}_i - \mathbf{f}_i^e.$$

These formulae are obtained by examining the relationship between \mathbf{f}_i and \mathbf{a}_i in Fig. 2.7 and assuming that \mathbf{I}_j^A

Table 2.8 Articulated-body algorithm for forward dynamics

```

inputs:  $\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, model, {}^0\mathbf{f}_i^e$ 
output:  $\ddot{\mathbf{q}}$ 
model data:  $N_B, \text{jtype}(i), p(i), X_L(i), \mathbf{I}_i$ 

 $\mathbf{v}_0 = \mathbf{0}$ 
 $\mathbf{a}_0 = -\mathbf{a}_g$ 
for  $i = 1$  to  $N_B$  do
   $X_J(i) = \text{xjcalc}(\text{jtype}(i), \mathbf{q}_i)$ 
   ${}^iX_{p(i)} = X_J(i) X_L(i)$ 
  if  $p(i) \neq 0$  then
     ${}^iX_0 = {}^iX_{p(i)} {}^{p(i)}X_0$ 
  end
   $\Phi_i = \text{pcalc}(\text{jtype}(i), \mathbf{q}_i)$ 
   $\Phi_{c_i} = \text{pdcalc}(\text{jtype}(i), \mathbf{q}_i, \dot{\mathbf{q}}_i)$ 
   $\mathbf{v}_i = {}^iX_{p(i)} \mathbf{v}_{p(i)} + \Phi_i \dot{\mathbf{q}}_i$ 
   $\boldsymbol{\zeta}_i = \Phi_{c_i} \dot{\mathbf{q}}_i + \mathbf{v}_i \times \Phi_i \dot{\mathbf{q}}_i$ 
   $\mathbf{I}_i^A = \mathbf{I}_i$ 
   $\mathbf{p}_i^A = \mathbf{v}_i \times \mathbf{I}_i \mathbf{v}_i - {}^iX_0^{-T} {}^0\mathbf{f}_i^e$ 
end
for  $i = N_B$  to  $1$  do
   $\mathbf{U}_i = \mathbf{I}_i^A \Phi_i$ 
   $\mathbf{D}_i = (\Phi_i^T \mathbf{U}_i)^{-1}$ 
   $\mathbf{u}_i = \boldsymbol{\tau}_i - \mathbf{U}_i^T \boldsymbol{\zeta}_i - \Phi_i^T \mathbf{p}_i^A$ 
  if  $p(i) \neq 0$  then
     $\mathbf{I}_{p(i)}^A = \mathbf{I}_{p(i)}^A + {}^iX_{p(i)}^T (\mathbf{I}_i^A - \mathbf{U}_i \mathbf{D}_i \mathbf{U}_i^T) {}^iX_{p(i)}$ 
     $\mathbf{p}_{p(i)}^A = \mathbf{p}_{p(i)}^A + {}^iX_{p(i)}^T (\mathbf{p}_i^A + \mathbf{I}_i^A \boldsymbol{\zeta}_i + \mathbf{U}_i \mathbf{D}_i \mathbf{u}_i)$ 
  end
end
for  $i = 1$  to  $N_B$  do
   $\mathbf{a}_i = {}^iX_{p(i)} \mathbf{a}_{p(i)}$ 
   $\ddot{\mathbf{q}}_i = \mathbf{D}_i (\mathbf{u}_i - \mathbf{U}_i^T \mathbf{a}_i)$ 
   $\mathbf{a}_i = \mathbf{a}_i + \Phi_i \ddot{\mathbf{q}}_i + \boldsymbol{\zeta}_i$ 
end

```

and \mathbf{p}_j^A are already known for every $j \in c(i)$. For more details see [2.1, 8, 15, 24].

The complete algorithm is shown in Table 2.8. It is expressed in link coordinates, as per the RNEA in Table 2.6. It makes a total of three passes through the kinematic tree. The first pass iterates from the base out to the tips; it calculates the link velocities using (2.76), the velocity-product term $\boldsymbol{\zeta}_i = \Phi_i \dot{\mathbf{q}}_i$, and it initializes the variables \mathbf{I}_i^A and \mathbf{p}_i^A to the values \mathbf{I}_i and $\mathbf{p}_i (= \mathbf{v}_i \times \mathbf{I}_i \mathbf{v}_i - {}^iX_0^T {}^0\mathbf{f}_i^e)$, respectively. The second pass iterates from the tips back to the base; it calculates the articulated-body inertia and bias force for each link using (2.81) and (2.82). The third pass iterates from the base to the tips; it calculates the link and joint accelerations using (2.80) and (2.77).

2.5.3 The Composite-Rigid-Body Algorithm

The CRBA is an algorithm for calculating the joint-space inertia matrix (JSIM) of a kinematic tree. The most common use for the CRBA is as part of a forward dynamics algorithm. It first appeared as method 3 in [2.5].

Forward dynamics, in joint space, is the task of calculating $\ddot{\mathbf{q}}$ from $\mathbf{q}, \dot{\mathbf{q}}$, and $\boldsymbol{\tau}$. Starting from (2.37), the most obvious way to proceed is to calculate \mathbf{H} and $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$, and then solve the linear equation

$$\mathbf{H}\ddot{\mathbf{q}} = \boldsymbol{\tau} - (\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g) \quad (2.83)$$

for $\ddot{\mathbf{q}}$. If the mechanism is a kinematic tree, then \mathbf{H} and $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ can be computed in $O(n^2)$ and $O(n)$ operations, respectively, and (2.83) can be solved in $O(n^3)$ operations. Algorithms that take this approach are therefore known collectively as $O(n^3)$ algorithms. However, this figure of $O(n^3)$ should be regarded as the worst-case complexity, since the actual complexity depends on the amount of branching in the tree [2.25]. Furthermore, even in the worst case, the n^3 term has a small coefficient, and does not dominate until approximately $n = 60$.

$\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ can be calculated using an inverse dynamics algorithm. If $ID(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ is the result of an inverse dynamics calculation with arguments $\mathbf{q}, \dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$, then

$$ID(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \boldsymbol{\tau} = \mathbf{H}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g,$$

so

$$\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g = ID(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{0}). \quad (2.84)$$

Thus, the value of $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ for a kinematic tree can be calculated efficiently using the RNEA with $\ddot{\mathbf{q}} = \mathbf{0}$.

The key concept in the CRBA is to note that the JSIM only depends on the joint positions, and not their rates. The CRBA makes the simplifying assumption that the

rate at each joint is zero. By also assuming that gravity is zero, $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ is eliminated from (2.83). Furthermore, for a revolute joint, a unit of joint acceleration applied at the j -th joint produces the j -th column of the **JSIM**. This partitions the mechanism into two composite rigid bodies connected by the j -th joint, and simplifies the dynamics considerably. This concept has been generalized so that the **CRBA** may be applied to any joint type within a kinematic tree structure.

It can be shown that the general form of the **JSIM** for a kinematic tree is

$$\mathbf{H}_{ij} = \begin{cases} \boldsymbol{\Phi}_i^T \mathbf{I}_i^C \boldsymbol{\Phi}_j & \text{if } i \in c^*(j) \\ \boldsymbol{\Phi}_i^T \mathbf{I}_j^C \boldsymbol{\Phi}_j & \text{if } j \in c^*(i) \\ \mathbf{0} & \text{otherwise,} \end{cases} \quad (2.85)$$

where $c^*(i)$ is the set of links in the subtree rooted at link i , including link i itself, and

$$\mathbf{I}_i^C = \sum_{j \in c^*(i)} \mathbf{I}_j. \quad (2.86)$$

Table 2.9 Composite-rigid-body algorithm for calculating the **JSIM**

inputs: *model*, *RNEA* partial results

output: *H*

model data: N_B , $p(i)$, \mathbf{I}_i

RNEA data: $\boldsymbol{\Phi}_i$, ${}^i\mathbf{X}_{p(i)}$

$\mathbf{H} = \mathbf{0}$

for $i = 1$ **to** N_B **do**

$\mathbf{I}_i^C = \mathbf{I}_i$

end

for $i = N_B$ **to** 1 **do**

$\mathbf{F} = \mathbf{I}_i^C \boldsymbol{\Phi}_i$

$\mathbf{H}_{ii} = \boldsymbol{\Phi}_i^T \mathbf{F}$

if $p(i) \neq 0$ **then**

$\mathbf{I}_{p(i)}^C = \mathbf{I}_{p(i)}^C + {}^i\mathbf{X}_{p(i)}^T \mathbf{I}_i^C {}^i\mathbf{X}_{p(i)}$

end

$j = i$

while $p(j) \neq 0$ **do**

$\mathbf{F} = {}^j\mathbf{X}_{p(j)}^T \mathbf{F}$

$j = p(j)$

$\mathbf{H}_{ij} = \mathbf{F}^T \boldsymbol{\Phi}_j$

$\mathbf{H}_{ji} = \mathbf{H}_{ij}^T$

end

end

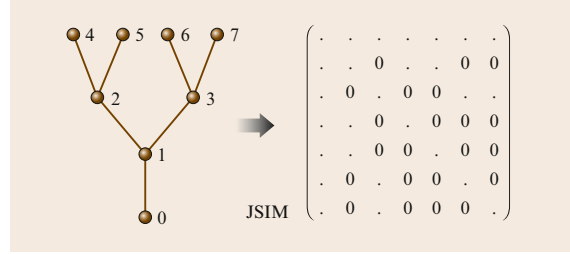


Fig. 2.8 Branch-induced sparsity: branches in the kinematic tree cause certain elements in the **JSIM** to be zero

See [2.8, 15]. In fact, \mathbf{I}_i^C is the inertia of the composite rigid body formed by the rigid assembly of all the links in $c^*(i)$, and this is where the algorithm gets its name.

Equations (2.85) and (2.86) are the basis of the algorithm shown in Table 2.9, which is the **CRBA** in link coordinates. This algorithm assumes that the matrices ${}^i\mathbf{X}_{p(i)}$ and $\boldsymbol{\Phi}_i$ have already been calculated, e.g., during the calculation of $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$. If this is not the case, then the relevant lines from Table 2.6 can be inserted into the first loop. The matrix \mathbf{F} is a local variable. The first step, $\mathbf{H} = \mathbf{0}$, can be omitted if there are no branches in the tree.

Having calculated $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ and \mathbf{H} , the final step is to solve (2.83) for $\ddot{\mathbf{q}}$. This can be done using a standard Cholesky or \mathbf{LDL}^T factorization. Note that \mathbf{H} can be highly ill-conditioned [2.26], reflecting an underlying ill-conditioning of the kinematic tree itself, so it is recommended to use double-precision arithmetic for every step in the forward dynamics calculation. (This advice applies also to the **ABA**.)

Exploiting Sparsity

Equation (2.85) implies that some elements of \mathbf{H} will automatically be zero if there are branches in the kinematic tree. An example of this effect is shown in Fig. 2.8. Observe that nearly half of the elements are zero. It is possible to exploit this sparsity using the factorization algorithms described in [2.25]. Depending on the amount of branching in the tree, the sparse algorithms can run many times faster than the standard algorithms.

2.5.4 Operational-Space Inertia Matrix

Two different algorithms are presented to calculate the **OSIM**. The first is an $O(n^3)$ algorithm that uses the basic definition of the **OSIM** along with efficient factorization of the **JSIM**. The second is an $O(n)$ algorithm that is based on efficient solution of the forward dynamics problem.

Algorithm Using Basic Definition

If a robot has a relatively small number of freedoms (e.g., six), then the most efficient method for calculating the **OSIM** is via (2.52). The procedure is as follows.

1. Calculate \mathbf{H} via the **CRBA**.
2. Factorize \mathbf{H} into $\mathbf{H} = \mathbf{L}\mathbf{L}^T$ (Cholesky factorization).
3. Use back-substitution to calculate $\mathbf{Y} = \mathbf{L}^{-1} \mathbf{J}^T$.
4. $\mathbf{A}^{-1} = \mathbf{Y}^T \mathbf{Y}$.
5. Factorize \mathbf{A}^{-1} (optional).

The final step is only possible if the end-effector has a full six **DOFs**, and is only necessary if the application requires \mathbf{A} rather than \mathbf{A}^{-1} . In the second step, an **LDL^T** factorization can be used instead of Cholesky, or one can use one of the efficient factorizations described in [2.25] for branched kinematic trees.

The other terms in (2.38) can be calculated via (2.53) and (2.54). In particular, (2.38) can be rewritten in the form

$$\dot{\mathbf{v}} + \mathbf{A}^{-1}(\mathbf{x})[\boldsymbol{\mu}(\mathbf{x}, \mathbf{v}) + \boldsymbol{\rho}(\mathbf{x})] = \mathbf{A}^{-1}(\mathbf{x}) \mathbf{f}, \quad (2.87)$$

and the quantity $\mathbf{A}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho})$ can be calculated from the formula

$$\mathbf{A}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho}) = \mathbf{J}\mathbf{H}^{-1}(\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g) - \dot{\mathbf{J}}\dot{\mathbf{q}}. \quad (2.88)$$

The term $\dot{\mathbf{J}}\dot{\mathbf{q}}$ is the velocity-product acceleration of the end-effector (2.50). It is calculated as a by-product of calculating $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ via the **RNEA** (2.84). Specifically, $\dot{\mathbf{J}}\dot{\mathbf{q}} = \mathbf{a}_{ee} - \mathbf{a}_0$, where \mathbf{a}_{ee} is the calculated acceleration of the end-effector (expressed in the same coordinates as $\dot{\mathbf{v}}$) and \mathbf{a}_0 is the acceleration of the base ($-\mathbf{a}_g$).

$O(n)$ Algorithm

For a sufficiently large value of n , it becomes more efficient to use an $O(n)$ algorithm. Several such algorithms can be found in [2.27–29]. In this section, a more straightforward algorithm is given, which is based on an $O(n)$ calculation of the joint-space forward dynamics problem, e.g., via the **ABA**. It is a variation of the unit force method [2.28] and computes the inverse of the **OSIM**.

Starting with (2.87), observe that \mathbf{A}^{-1} is a function of position only, and certain terms in the dynamic equations can be neglected without affecting its value. Specifically, if the joint rates, $\dot{\mathbf{q}}$, joint forces, $\boldsymbol{\tau}$, and gravitational forces are all set to zero, the value of \mathbf{A} will remain unchanged. Under these conditions,

$$\dot{\mathbf{v}} = \mathbf{A}^{-1} \mathbf{f}. \quad (2.89)$$

Let us define $\hat{\mathbf{e}}_i$ to be a 6-D coordinate vector with a 1 in the i -th coordinate and zeros elsewhere. If we set

$\mathbf{f} = \hat{\mathbf{e}}_i$ in (2.89), then $\dot{\mathbf{v}}$ will equal column i of \mathbf{A}^{-1} . Let us also define the function $FD(i, j, \mathbf{q}, \dot{\mathbf{q}}, \mathbf{a}_0, \boldsymbol{\tau}, \mathbf{f})$, which performs a forward-dynamics calculation and returns the true acceleration of link i (i.e., $\mathbf{a}_i - \mathbf{a}_0$), expressed in the same coordinates as \mathbf{f} (typically the base coordinates). The arguments \mathbf{q} , $\dot{\mathbf{q}}$ and $\boldsymbol{\tau}$ set the values of the joint position, velocity, and force variables, while j and \mathbf{f} specify that an external force of \mathbf{f} is to be applied to link j . The argument \mathbf{a}_0 specifies a fictitious base acceleration to include gravitational effects, and is set to either $\mathbf{0}$ or $-\mathbf{a}_g$.

With these definitions, we have

$$(\mathbf{A}^{-1})^i = FD(ee, ee, \mathbf{q}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \hat{\mathbf{e}}_i) \quad (2.90)$$

and

$$\mathbf{A}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho}) = -FD(ee, ee, \mathbf{q}, \dot{\mathbf{q}}, -\mathbf{a}_g, \boldsymbol{\tau}, \mathbf{0}), \quad (2.91)$$

where $(\mathbf{A}^{-1})^i$ is column i of \mathbf{A}^{-1} , and ee is the body number of the end-effector. It therefore follows that the coefficients of (2.87) can be calculated using the algorithm shown in Table 2.10. This algorithm is $O(n)$.

The efficiency of the algorithm may be increased significantly when computing \mathbf{A}^{-1} by noting that: (1) \mathbf{v}_i , $\boldsymbol{\zeta}_i$, and $\boldsymbol{\tau}_i$ in the **ABA** calculation (Table 2.8) may be set to zero, and (2) \mathbf{I}_i^A and the quantities that depend upon it (\mathbf{U}_i and \mathbf{D}_i) need only be computed once, since they do not vary with the applied force. Also, note that the algorithm may be applied to multiple end-effectors by modifying FD to accept a list of end-effector body numbers in its first argument, and return a composite vector containing the accelerations of all the specified bodies. The algorithm in Table 2.10 is then enclosed in a **for** loop that controls the second argument to FD , and iterates over all the end-effector body numbers [2.19]. However, with several end-effectors, the force propagation method of *Lilly* [2.28] should probably be used for enhanced computational efficiency. If there are m end-effectors, then the complexity is $O(mn)$.

Table 2.10 Algorithm to compute the inverse of the operational-space inertia matrix and other terms

for $j = 1$ to 6 **do**

$\dot{\mathbf{v}}^j = FD(ee, ee, \mathbf{q}, \mathbf{0}, \mathbf{0}, \mathbf{0}, \hat{\mathbf{e}}_j)$

end

$\mathbf{A}^{-1} = [\dot{\mathbf{v}}^1 \ \dot{\mathbf{v}}^2 \ \dots \ \dot{\mathbf{v}}^6]$

$\mathbf{A}^{-1}(\boldsymbol{\mu} + \boldsymbol{\rho}) = -FD(ee, ee, \mathbf{q}, \dot{\mathbf{q}}, -\mathbf{a}_g, \boldsymbol{\tau}, \mathbf{0})$

2.6 Kinematic Loops

All of the algorithms in the last section were for kinematic trees. In this section, a final algorithm is provided for the forward dynamics of closed-loop systems. The algorithm supplements the dynamic equations of motion for a spanning tree of the closed-loop system, with the loop-closure constraint equations. Three different methods are given to solve the resulting linear system of equations. An efficient algorithm is given to compute the loop-closure constraints.

Systems with closed kinematic loops exhibit more complicated dynamics than kinematic trees. For example:

1. The degree of motion freedom of a kinematic tree is fixed, but that of a closed-loop system can vary.
2. The degree of instantaneous motion freedom is always the same as the degree of finite motion freedom in a kinematic tree, but they can be different in a closed-loop system.
3. Every force in a kinematic tree can be determined, but some forces in a closed-loop system can be indeterminate. This occurs whenever a closed-loop system is overconstrained.

Two examples of these phenomena are shown in Fig. 2.9. The mechanism in Fig. 2.9a has no finite motion freedom, but it has two degrees of infinitesimal motion freedom. The mechanism in Fig. 2.9b has one degree of freedom when $\theta \neq 0$, but if $\theta = 0$ then the two arms, *A* and *B*, are able to move independently, and the mechanism has two degrees of freedom. Moreover, at the boundary between these two motion regimes, the mechanism has three degrees of infinitesimal motion freedom. Both these mechanisms are planar, and are therefore overconstrained.

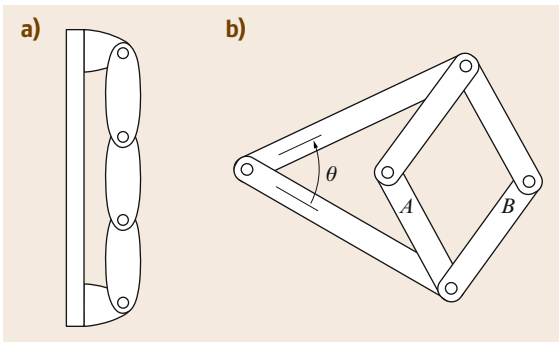


Fig. 2.9 Pathological closed-loop systems

As a result, the out-of-plane components of the joint constraint forces are indeterminate. This kind of indeterminacy has no effect on the motions of these mechanisms, but it does complicate the calculation of their dynamics.

2.6.1 Formulation of Closed-Loop Algorithm

A closed-loop system can be modeled as a spanning tree subject to a set of loop-closure constraint forces. If

$$H\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g = \boldsymbol{\tau}$$

is the equation of motion of the spanning tree on its own, then the equation of motion for the closed-loop system is

$$H\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g = \boldsymbol{\tau} + \boldsymbol{\tau}^a + \boldsymbol{\tau}^c, \quad (2.92)$$

where $\boldsymbol{\tau}^a$ and $\boldsymbol{\tau}^c$ are vectors of loop-closure active and constraint forces, respectively, expressed in the generalized force coordinates of the spanning tree. $\boldsymbol{\tau}^a$ is a known quantity, and $\boldsymbol{\tau}^c$ is unknown. $\boldsymbol{\tau}^a$ comes from the force elements acting at the loop-closing joints (springs, dampers and actuators). If there are no such force elements, then $\boldsymbol{\tau}^a = \mathbf{0}$.

The loop-closure constraints restrict the motion of the spanning tree. At the acceleration level, these constraints can be expressed in the form of a linear equation,

$$\mathbf{L}\ddot{\mathbf{q}} = \mathbf{1}, \quad (2.93)$$

where \mathbf{L} is an $n^c \times n$ matrix. n^c is the number of constraints due to the loop-closing joints, and is given by the formula

$$n^c = \sum_{k=N_B+1}^{N_J} n_k^c, \quad (2.94)$$

where n_k^c is the number of constraints imposed by joint *k*. If $\text{rank}(\mathbf{L}) < n^c$ then the loop-closure constraints are linearly dependent, and the closed-loop mechanism is overconstrained. The mobility of a closed-loop system (i.e., its degree of motion freedom) is given by the formula

$$\text{mobility} = n - \text{rank}(\mathbf{L}). \quad (2.95)$$

Given a constraint equation in the form of (2.93), it follows that the constraint forces can be expressed in the form

$$\boldsymbol{\tau}^c = \mathbf{L}^T \boldsymbol{\lambda}, \quad (2.96)$$

where $\boldsymbol{\lambda} = (\lambda_{N_B+1}^T \cdots \lambda_{N_J}^T)^T$ is an $n^c \times 1$ vector of unknown constraint-force variables (or Lagrange multipliers). If the mechanism is overconstrained, then \mathbf{L}^T will have a null space, and the component of $\boldsymbol{\lambda}$ lying in this null space will be indeterminate.

It is often possible to identify redundant constraints in advance. For example, if a kinematic loop is known to be planar, then the out-of-plane loop-closure constraints are redundant. In these circumstances, it is advantageous to remove the corresponding rows of \mathbf{L} and elements of \mathbf{I} and $\boldsymbol{\lambda}$. The removed elements of $\boldsymbol{\lambda}$ can be assigned a value of zero.

Combining (2.92), (2.93), and (2.96) produces the following equation of motion for a closed-loop system:

$$\begin{pmatrix} \mathbf{H} & \mathbf{L}^T \\ \mathbf{L} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \ddot{\mathbf{q}} \\ -\boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\tau} + \boldsymbol{\tau}^a - (\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g) \\ \mathbf{I} \end{pmatrix}. \quad (2.97)$$

The system matrix is symmetric, but indefinite. If \mathbf{L} has full rank, then the system matrix will be nonsingular, otherwise it will be singular, and one or more elements of $\boldsymbol{\lambda}$ will be indeterminate.

Equation (2.97) can be solved in any of the following ways:

1. solve it directly for $\ddot{\mathbf{q}}$ and $\boldsymbol{\lambda}$
2. solve for $\boldsymbol{\lambda}$ first, and then use the result to solve for $\ddot{\mathbf{q}}$
3. solve (2.93) for $\ddot{\mathbf{q}}$, substitute the result into (2.92), eliminate the unknown constraint forces, and solve for the remaining unknowns

Method 1 is the simplest, but generally also the least efficient. This method is appropriate when the system matrix is nonsingular. As the size of the system matrix is $(n + n^c) \times (n + n^c)$, this method is $O((n + n^c)^3)$.

Method 2 is particularly useful if $n \gg n^c$, and offers the opportunity to use $O(n)$ algorithms on the spanning tree [2.6]. From (2.97),

$$\mathbf{L}\mathbf{H}^{-1}\mathbf{L}^T\boldsymbol{\lambda} = \mathbf{I} - \mathbf{L}\mathbf{H}^{-1}[\boldsymbol{\tau} + \boldsymbol{\tau}^a - (\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g)]. \quad (2.98)$$

This equation can be formulated in $O(n(n^c)^2)$ operations via $O(n)$ algorithms, and solved in $O((n^c)^3)$. Once $\boldsymbol{\lambda}$ is known, $\boldsymbol{\tau}^c$ can be calculated via (2.96) in $O(nn^c)$ operations, and (2.92) solved by an $O(n)$ al-

gorithm; so the total complexity is $O(n(n^c)^2 + (n^c)^3)$. If \mathbf{L} is rank deficient, then $\mathbf{L}\mathbf{H}^{-1}\mathbf{L}^T$ will be singular; but it is still a positive-semidefinite matrix, and presents a slightly easier factorization problem than a singular instance of the indefinite system matrix in (2.97).

Method 3 is useful if $n - n^c$ is small, or if \mathbf{L} is expected to be rank deficient. Equation (2.93) is solved using a special version of Gaussian elimination (or similar procedure), which is equipped with a numerical rank test, and which is designed to solve underdetermined systems. The solution is an equation of the form

$$\ddot{\mathbf{q}} = \mathbf{K}\mathbf{y} + \ddot{\mathbf{q}}_0,$$

where $\ddot{\mathbf{q}}_0$ is any particular solution to (2.93), \mathbf{K} is an $n \times (n - \text{rank}(\mathbf{L}))$ matrix with the property $\mathbf{L}\mathbf{K} = \mathbf{0}$, and \mathbf{y} is a vector of $n - \text{rank}(\mathbf{L})$ unknowns. (Typically, \mathbf{y} is a linearly independent subset of the elements of $\ddot{\mathbf{q}}$.) Substituting this expression for $\ddot{\mathbf{q}}$ into (2.92), and pre-multiplying both sides by \mathbf{K}^T to eliminate $\boldsymbol{\tau}^c$, produces

$$\mathbf{K}^T \mathbf{H} \mathbf{K} \mathbf{y} = \mathbf{K}^T (\boldsymbol{\tau} + \boldsymbol{\tau}^a - (\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g) - \mathbf{H}\ddot{\mathbf{q}}_0). \quad (2.99)$$

This method also has cubic complexity, but it can be the most efficient if $n - n^c$ is small. It is also reported to be more stable than method 1 [2.30].

2.6.2 Closed-Loop Algorithm

Algorithms for calculating \mathbf{H} and $\mathbf{C}\dot{\mathbf{q}} + \boldsymbol{\tau}_g$ can be found in Sections 2.5.3 and 2.5.1, respectively, which just leaves \mathbf{L} , \mathbf{I} and $\boldsymbol{\tau}^a$. To keep things simple, we will assume that all loop-closing joints are zero-DOF joints. There is no loss of generality with this assumption: one simply breaks open the loops by cutting links instead of joints (see Fig. 2.10). However, there may be some loss of efficiency. With this assumption, we only need to calculate \mathbf{L} and \mathbf{I} , since $\boldsymbol{\tau}^a = \mathbf{0}$.

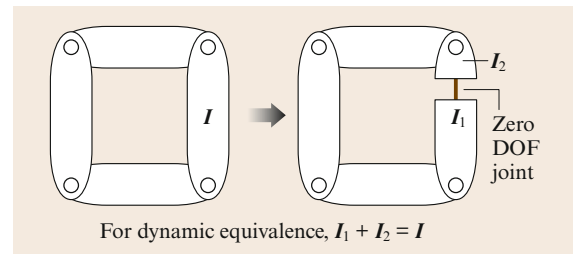


Fig. 2.10 Inserting a zero-DOF joint in preparation for cutting the loop open at that joint

Loop Constraints

In the general case, the velocity constraint equation for loop k is

$$(\Psi_k^c)^T (\mathbf{v}_{s(k)} - \mathbf{v}_{p(k)}) = \mathbf{0}, \quad (2.100)$$

and the acceleration constraint is

$$(\Psi_k^c)^T (\mathbf{a}_{s(k)} - \mathbf{a}_{p(k)}) + (\Psi_k^c)^T (\mathbf{v}_{s(k)} - \mathbf{v}_{p(k)}) = \mathbf{0}. \quad (2.101)$$

However, if every loop-closing joint has zero DOF, then these equations simplify to

$$\mathbf{v}_{s(k)} - \mathbf{v}_{p(k)} = \mathbf{0} \quad (2.102)$$

and

$$\mathbf{a}_{s(k)} - \mathbf{a}_{p(k)} = \mathbf{0}. \quad (2.103)$$

Let us define a loop Jacobian, \mathbf{J}_k , with the property that

$$\mathbf{v}_{s(k)} - \mathbf{v}_{p(k)} = \mathbf{J}_k \dot{\mathbf{q}}. \quad (2.104)$$

\mathbf{J}_k is a $6 \times n$ matrix defined by the formula

$$\mathbf{J}_k = (e_{1k} \Phi_1 \cdots e_{N_B k} \Phi_{N_B}), \quad (2.105)$$

where

$$e_{ik} = \begin{cases} +1 & \text{if } s(k) \in c^*(i) \text{ and } p(k) \notin c^*(i), \\ -1 & \text{if } p(k) \in c^*(i) \text{ and } s(k) \notin c^*(i), \\ 0 & \text{otherwise.} \end{cases}$$

In other words, $e_{ik} = +1$ if joint i lies on the path to $s(k)$ but not the path to $p(k)$; $e_{ik} = -1$ if joint i lies on the path to $p(k)$ but not the path to $s(k)$; and $e_{ik} = 0$ if joint i lies on both paths or on neither.

The loop acceleration constraint can now be written

$$\begin{aligned} \mathbf{0} &= \mathbf{a}_{s(k)} - \mathbf{a}_{p(k)} \\ &= \mathbf{J}_k \ddot{\mathbf{q}} + \dot{\mathbf{J}}_k \dot{\mathbf{q}} \\ &= \mathbf{J}_k \ddot{\mathbf{q}} + \mathbf{a}_{s(k)}^{\text{vp}} - \mathbf{a}_{p(k)}^{\text{vp}}, \end{aligned} \quad (2.106)$$

where \mathbf{a}_i^{vp} is the velocity-product acceleration of link i , which is the acceleration it would have if $\ddot{\mathbf{q}}$ were zero. The velocity-product acceleration of every link is calculated during the calculation of the vector $\mathbf{C}\dot{\mathbf{q}} + \mathbf{t}_g$ (2.84). If the RNEA is used to calculate $\mathbf{C}\dot{\mathbf{q}} + \mathbf{t}_g$, then

\mathbf{a}_i^{vp} will be the value of \mathbf{a}_i calculated by the RNEA with its acceleration argument set to zero.

The matrices \mathbf{L} and \mathbf{I} can now be expressed as follows:

$$\mathbf{L} = \begin{pmatrix} \mathbf{L}_{N_B+1} \\ \vdots \\ \mathbf{L}_{N_J} \end{pmatrix} \quad \text{and} \quad \mathbf{I} = \begin{pmatrix} \mathbf{I}_{N_B+1} \\ \vdots \\ \mathbf{I}_{N_J} \end{pmatrix}, \quad (2.107)$$

where

$$\mathbf{L}_k = \mathbf{J}_k \quad (2.108)$$

and

$$\mathbf{I}_k = \mathbf{a}_{p(k)}^{\text{vp}} - \mathbf{a}_{s(k)}^{\text{vp}}. \quad (2.109)$$

Constraint Stabilization

In practice, it is necessary to stabilize loop-closure constraints, or they will simply fly apart during simulation because of numerical integration errors. The standard technique is due to *Baumgarte* [2.3, 7, 31], and consists of replacing each constraint equation of the form

$$a_e = 0$$

with one of the form

$$a_e + K_v v_e + K_p p_e = 0,$$

where a_e , v_e , and p_e are the acceleration, velocity, and position errors, respectively, and K_v and K_p are positive constants. Typically, one chooses a time constant, t_c , according to how quickly one wants the position and velocity errors to decay. K_p and K_v are then given by the formulae $K_v = 2/t_c$ and $K_p = 1/t_c^2$. However, there is no good rule for choosing t_c . If t_c is too long, then loop-constraint errors accumulate faster than they decay; if t_c is too short, then the equations of motion become excessively stiff, causing a loss of numerical integration accuracy. A reasonable value for a large, slow industrial robot is $t_c = 0.1$, while a smaller, faster robot might need $t_c = 0.01$.

To incorporate stabilization terms into the loop constraint equation, we replace (2.109) with

$$\mathbf{I}_k = \mathbf{a}_{p(k)}^{\text{vp}} - \mathbf{a}_{s(k)}^{\text{vp}} - K_v (\mathbf{v}_{s(k)} - \mathbf{v}_{p(k)}) - K_p \mathbf{p}_{ek}, \quad (2.110)$$

where \mathbf{p}_{ek} is a vector representing the position error in loop k . In absolute coordinates (i. e., link 0 coordinates), \mathbf{p}_{ek} is given by

$$\mathbf{p}_{ek} = \mathbf{x_to_vec}(^0X_{p(k)} X_{L1}^{-1}(k) X_{L2}(k)^{s(k)} X_0), \quad (2.111)$$

where the $X_{L1}(k)$ and $X_{L2}(k)$ transforms are defined in (2.62) and (2.63), and shown in Fig. 2.4 for joint k , and $\mathbf{x_to_vec}(^B X_A)$ computes a vector approximating the displacement from frame A to frame B , assuming this displacement to be infinitesimal. $\mathbf{x_to_vec}$ can be defined as

$$\mathbf{x_to_vec}(X) = \frac{1}{2} \begin{pmatrix} X_{23} - X_{32} \\ X_{31} - X_{13} \\ X_{12} - X_{21} \\ X_{53} - X_{62} \\ X_{61} - X_{43} \\ X_{42} - X_{51} \end{pmatrix}. \quad (2.112)$$

Algorithm

Table 2.11 shows an algorithm for calculating \mathbf{L} and \mathbf{l} for the special case when all the loop-closing joints have zero DOF. It combines simplicity with good performance by transforming every quantity that is needed to formulate the loop-closure constraints into a single coordinate system, in this case absolute (link 0) coordinates, so that no further transforms are needed.

The first loop calculates the transforms from absolute to link coordinates, and uses them to transform Φ_i to absolute coordinates. Only the Φ_i that are needed in the loop-closure constraints are transformed.

The second loop calculates the nonzero elements of \mathbf{L} (which can be sparse), according to (2.105). The inner while loop terminates on the root of the loop, which is the highest-numbered common ancestor of links $p(k)$ and $s(k)$. It could be the fixed base if they have no other common ancestor. The second loop ends with the calculation of \mathbf{l} , in absolute coordinates, according to (2.110).

Table 2.11 Algorithm to calculate loop-closure constraints

```

inputs: model, RNEA partial results
outputs:  $\mathbf{L}$ ,  $\mathbf{l}$ 
model data:  $N_B$ ,  $p(i)$ ,  $N_J$ ,  $p(k)$ ,  $s(k)$ ,  $LB(i)$ ,
 $X_{L1}(k)$ ,  $X_{L2}(k)$ ,  $K_p$ ,  $K_v$ 
RNEA data:  $\Phi_i$ ,  ${}^iX_{p(i)}$ ,  $\mathbf{v}_{p(k)}$ ,  $\mathbf{v}_{s(k)}$ ,  $\mathbf{a}_{p(k)}^{vp}$ ,  $\mathbf{a}_{s(k)}^{vp}$ 
for  $i = 1$  to  $N_B$  do
  if  $p(i) \neq 0$  then
     ${}^iX_0 = {}^iX_{p(i)} {}^{p(i)}X_0$ 
  end
  if  $LB(i) \neq \text{null}$  then
     ${}^0\Phi_i = {}^iX_0^{-1} \Phi_i$ 
  end
end
 $\mathbf{L} = \mathbf{0}$ 
for  $k = N_B + 1$  to  $N_J$  do
   $i = p(k)$ 
   $j = s(k)$ 
  while  $i \neq j$  do
    if  $i > j$  then
       $L_{k,i} = -{}^0\Phi_i$ 
       $i = p(i)$ 
    else
       $L_{k,j} = {}^0\Phi_j$ 
       $j = p(j)$ 
    end
  end
   $\mathbf{a}_e = {}^{s(k)}X_0^{-1} \mathbf{a}_{s(k)}^{vp} - {}^{p(k)}X_0^{-1} \mathbf{a}_{p(k)}^{vp}$ 
   $\mathbf{v}_e = {}^{s(k)}X_0^{-1} \mathbf{v}_{s(k)} - {}^{p(k)}X_0^{-1} \mathbf{v}_{p(k)}$ 
   $\mathbf{p}_e = \mathbf{x\_to\_vec}({}^{p(k)}X_0^{-1} X_{L1}^{-1}(k) X_{L2}(k)^{s(k)} X_0)$ 
   $\mathbf{l}_k = -\mathbf{a}_e - K_v \mathbf{v}_e - K_p \mathbf{p}_e$ 
end

```

2.7 Conclusions and Further Reading

This chapter has presented the fundamentals of rigid-body dynamics as they apply to robot mechanisms. It has covered the following topics: the spatial vector algebra, which provides a concise notation for describing and implementing dynamics equations and algorithms; the canonical equations of motion that are most frequently used in robotics; how to construct a dynamic model of

a robot; and several efficient model-based algorithms for calculating inverse dynamics, forward dynamics, and the joint-space and operational-space inertia matrices.

There are many topics in dynamics that have not been mentioned in this chapter, but can be found in later chapters of this handbook. The dynamics of robots with elastic links and joints is covered in Chap. 13; the prob-

lem of identifying the parameters of a dynamic model is covered in Chap. 14; the dynamics of physical contact between a robot and the objects in its environment is described in Chap. 27; and the dynamics of robots with floating bases is described in Chap. 45.

We conclude this chapter by noting that a brief history of robot dynamics can be found in [2.32], and that a more extensive treatment of robot dynamics can be found in books such as [2.8, 10, 15, 28, 33–35]. Finally, some suggestions for further reading are listed below.

2.7.1 Multibody Dynamics

Robot dynamics can be regarded as a subset (or a specific application) of the broader discipline of multibody dynamics. Books on multibody dynamics include [2.3, 14, 31, 36–41]. Of course, multibody dynamics is, in turn, a subset of classical mechanics; and the mathematical foundations of the subject can be found in any good book on classical mechanics, such as [2.13].

2.7.2 Alternative Representations

This chapter has used spatial vectors to express the equations of motion. There are various alternatives to the use of spatial vectors: other kinds of 6-D vector, 3-D vectors, 4×4 matrices, and the spatial operator algebra. All 6-D vector formalisms are similar, but are not exactly the same. The main alternatives to spatial vectors are: screws [2.10–12], motors [2.42], Lie algebras [2.12, 43], and ad hoc notations. (An ad hoc notation is one in which 3-D vectors are grouped into pairs for the purpose of reducing the volume of algebra.) Three-dimensional vectors are the formalism used in most classical mechanics and multibody texts, and are also a precursor to 6-D vector and 4×4 matrix formalisms. 4×4 matrices are popular in robotics because they are very useful for kinematics. However, they are not so useful for dynamics. 4×4 matrix formulations of dynamics can be found in [2.33, 44, 45]. The spatial operator algebra was developed at the Jet Propulsion Laboratory (JPL) by Rodriguez, Jain, and others. It uses $6N$ -dimensional vectors and $6N \times 6N$ matrices, the latter regarded as linear operators. Examples of this notation can be found in [2.46–48].

2.7.3 Alternative Formulations

This chapter used a vectorial formulation of the equations of motion that is usually called the Newtonian or Newton–Euler formulation. The main alternative is

the Lagrangian formulation, in which the equations of motion are obtained via Lagrange’s equation. Examples of the Lagrangian formulation can be found in [2.9, 10, 17, 49, 50]. *Kane’s* method has also been applied in robotics [2.51, 52].

2.7.4 Efficiency

Because of the need for real-time implementation, especially in control, the robotics community has focused on the problem of computational efficiency. For *inverse dynamics*, the $O(n)$ recursive Newton–Euler algorithm (RNEA) of Luh et al. [2.4] remains the most important algorithm. Further improvements to the algorithm are given in [2.53, 54]. For *forward dynamics*, the two algorithms presented in this chapter remain the most important for computational considerations: the $O(n)$ articulated-body algorithm (ABA) developed by Featherstone [2.1] and the $O(n^3)$ algorithm based on the composite-rigid-body algorithm (CRBA) of Walker and Orin [2.5]. Improvements were made in the ABA over the years [2.15, 16, 24] so that it was more efficient than the CRBA-based algorithm for decreasingly smaller values of n . However, recent application of the CRBA to branched kinematic trees [2.25] and robotic systems with motion-controlled appendages [2.55] continue to show the viability of the CRBA approach.

For the *joint-space inertia matrix*, the CRBA [2.5] is the most important algorithm. A number of improvements and modifications have been made over the years to increase its computational efficiency [2.15, 56–58]. For the *operational-space inertia matrix*, efficient $O(n)$ algorithms have been developed [2.27–29] and applied to increasingly complex systems [2.19].

2.7.5 Accuracy

Concerns can arise over the numerical accuracy of a dynamics algorithm, the accuracy of a simulation (i.e., numerical integration accuracy), or the accuracy of a dynamic model. The numerical accuracy of dynamics algorithms has received relatively little attention compared with efficiency. The RNEA, CRBA, and ABA have all been tested for accuracy on a large variety of rigid-body systems, but the same cannot be said of most other algorithms. Rigid-body systems are often ill-conditioned, in the sense that a small change in the applied force (or a model parameter) can produce a large change in the resulting acceleration. This phenomenon was studied by Featherstone [2.26], who discovered that the ill-conditioning gets worse with in-

creasing body count, and that it can grow in proportion to $O(n^4)$ in the worst case. Other publications on this topic include [2.8, 30, 59, 60].

2.7.6 Software Packages

A number of software packages have been developed to provide dynamic simulation capabilities for multibody systems, and in particular, robotic systems. Several have been written in MATLAB® for ease of integration with other analysis, control, and simulation programs. While many of these are open source, some are offered at a relatively low cost to the user. They differ in their capabilities in a variety of ways including: speed, topologies and joint models supported, accuracy, underlying dynamic formulation and associated order of complexity, user interface, graphics support, numerical integration routines, integration with other code, application support, and cost. Among those commonly cited are: Adams [2.61], Autolev [2.62], DynaMechs [2.63], Open Dynamics Engine [2.64], Robotics Studio [2.65], Robotics Toolbox [2.66], SD/FAST [2.67], SimMechanics [2.68], and Webots [2.69].

2.7.7 Symbolic Simplification

The technique of symbolic simplification takes a general-purpose dynamics algorithm, and applies it symbolically to a specific dynamic model. The result is a list of assignment statements detailing what the algorithm would have done if it had been executed for real. This list is then inspected and pruned of all unnecessary calculations, and the remainder is output to a text file in the form of computer source code. This code can run as much as ten times faster than the original general-purpose algorithm, but it is specific to one dynamic model. Both Autolev [2.62] and SD/FAST [2.67] use this technique. Other publications on symbolic simplification for dynamics include [2.70–75].

2.7.8 Algorithms for Parallel Computers

In order to speed up the common dynamics computations, a number of algorithms have been developed for parallel and pipelined computers. For *inverse dynamics*, early work focused on speeding up the $O(n)$ RNEA on up to n processors [2.76, 77] while subsequent work resulted in $O(\log_2 n)$ algorithms [2.78, 79]. For the $O(n^2)$ CRBA to compute the *joint-space inertia matrix*, early work resulted in $O(\log_2 n)$ algorithms for n processors to compute the composite-rigid-body inertias and diagonal elements of the matrix [2.80, 81]. Subsequent work resulted in $O(\log_2 n)$ algorithms for $O(n^2)$ processors to compute the entire matrix [2.82, 83]. For *forward dynamics*, speedup was obtained for a multiple manipulator system on a parallel/pipelined supercomputer [2.84]. The first $O(\log_2 n)$ algorithm for n processors was developed for an unbranched serial chain [2.85]. More recent work has focused on $O(\log_2 n)$ algorithms for more complex structures [2.60, 86, 87].

2.7.9 Topologically-Varying Systems

There are many robot mechanisms whose topology varies over time because of a change of contact conditions, especially with the environment. In legged vehicles, use of a compliant ground-contact model to compute the contact forces reduced the closed-loop structure to a tree structure [2.88]. However, for cases in which the contacts are very stiff, numerical integration problems may result. In more recent work [2.35, 89] in which hard contact constraints are assumed, an efficient method was used to reduce the large number of coordinate variables from that which may be necessary in general-purpose motion analysis systems [2.38]. Also, they were able to automatically identify the variables as the structure varied and developed a method for computing the velocity boundary conditions after configuration changes [2.35, 89].

References

- 2.1 R. Featherstone: The Calculation of Robot Dynamics using Articulated-Body Inertias, *Int. J. Robot. Res.* **2**(1), 13–30 (1983)
- 2.2 J.J. Craig: *Introduction to Robotics: Mechanics and Control*, 3rd edn. (Pearson Prentice Hall, Upper Saddle River, NJ 2005)
- 2.3 R.E. Roberson, R. Schwertassek: *Dynamics of Multi-body Systems* (Springer-Verlag, Berlin/Heidelberg/New York 1988)
- 2.4 J.Y.S. Luh, M.W. Walker, R.P.C. Paul: On-Line Computational Scheme for Mechanical Manipulators, *Trans. ASME J. Dyn. Syst. Measur. Control* **102**(2), 69–76 (1980)

- 2.5 M.W. Walker, D.E. Orin: Efficient Dynamic Computer Simulation of Robotic Mechanisms, *Trans. ASME J. Dyn. Syst. Measur. Control* **104**, 205–211 (1982)
- 2.6 D. Baraff: Linear-Time Dynamics using Lagrange Multipliers, *Proc. SIGGRAPH '96* (New Orleans 1996) pp. 137–146
- 2.7 J. Baumgarte: Stabilization of Constraints and Integrals of Motion in Dynamical Systems, *Comput. Methods Appl. Mech. Eng.* **1**, 1–16 (1972)
- 2.8 R. Featherstone: *Rigid Body Dynamics Algorithms* (Springer, Berlin, Heidelberg 2007)
- 2.9 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton, FL 1994)
- 2.10 J. Angeles: *Fundamentals of Robotic Mechanical Systems*, 2nd edn. (Springer-Verlag, New York 2003)
- 2.11 R.S. Ball: *A Treatise on the Theory of Screws* (Cambridge Univ. Press, London 1900), Republished (1998)
- 2.12 J.M. Selig: *Geometrical Methods in Robotics* (Springer, New York 1996)
- 2.13 D.T. Greenwood: *Principles of Dynamics* (Prentice-Hall, Englewood Cliffs, NJ 1988)
- 2.14 F.C. Moon: *Applied Dynamics* (Wiley, New York 1998)
- 2.15 R. Featherstone: *Robot Dynamics Algorithms* (Kluwer Academic, Boston 1987)
- 2.16 S. McMillan, D.E. Orin: Efficient Computation of Articulated-Body Inertias Using Successive Axial Screws, *IEEE Trans. Robot. Autom.* **11**, 606–611 (1995)
- 2.17 L. Sciavicco, B. Siciliano: *Modeling and Control of Robot Manipulators*, 2nd edn. (Springer, London 2000)
- 2.18 J. Slotine, W. Li: On the Adaptive Control of Robot Manipulators, *Int. J. Robot. Res.* **6**(3), 49–59 (1987)
- 2.19 K.S. Chang, O. Khatib: Operational Space Dynamics: Efficient Algorithms for Modeling and Control of Branching Mechanisms. In: *Proc. of IEEE International Conference on Robotics and Automation* (San Francisco 2000) pp. 850–856
- 2.20 O. Khatib: A Unified Approach to Motion and Force Control of Robot Manipulators: The Operational Space Formulation, *IEEE J. Robot. Autom.* **3**(1), 43–53 (1987)
- 2.21 Y.F. Zheng, H. Hemami: Mathematical Modeling of a Robot Collision with its Environment, *J. Robot. Syst.* **2**(3), 289–307 (1985)
- 2.22 W. Khalil, E. Dombre: *Modeling, Identification and Control of Robots* (Taylor & Francis, New York 2002)
- 2.23 J. Denavit, R.S. Hartenberg: A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices, *J. Appl. Mech.* **22**, 215–221 (1955)
- 2.24 H. Brandl, R. Johanni, M. Otter: A Very Efficient Algorithm for the Simulation of Robots and Similar Multibody Systems Without Inversion of the Mass Matrix. In: *Proc. of IFAC/IFIP/IMACS International Symposium on Theory of Robots*, (Vienna 1986)
- 2.25 R. Featherstone: Efficient Factorization of the Joint Space Inertia Matrix for Branched Kinematic Trees, *Int. J. Robot. Res.* **24**(6), 487–500 (2005)
- 2.26 R. Featherstone: An Empirical Study of the Joint Space Inertia Matrix, *Int. J. Robot. Res.* **23**(9), 859–871 (2004)
- 2.27 K. Kreutz-Delgado, A. Jain, G. Rodriguez: Recursive Formulation of Operational Space Control, *Proc. of IEEE International Conference on Robotics and Automation* (Sacramento, CA April 1991) pp. 1750–1753
- 2.28 K.W. Lilly: *Efficient Dynamic Simulation of Robotic Mechanisms* (Kluwer Academic, Norwell, MA 1993)
- 2.29 K.W. Lilly, D.E. Orin: Efficient O(N) Recursive Computation of the Operational Space Inertia Matrix, *IEEE Trans. Syst. Man Cybern.* **23**(5), 1384–1391 (1993)
- 2.30 R.E. Ellis, S.L. Ricker: Two Numerical Issues in Simulating Constrained Robot Dynamics, *IEEE Trans. Syst. Man Cybern.* **24**(1), 19–27 (1994)
- 2.31 J. Wittenburg: *Dynamics of Systems of Rigid Bodies* (B.G. Teubner, Stuttgart 1977)
- 2.32 R. Featherstone, D.E. Orin: Robot Dynamics: Equations and Algorithms. In: *Proc. of IEEE International Conference on Robotics and Automation*, (San Francisco, April 2000) pp. 826–834
- 2.33 C.A. Balafoutis, R.V. Patel: *Dynamic Analysis of Robot Manipulators: A Cartesian Tensor Approach* (Kluwer Academic, Boston 1991)
- 2.34 L.W. Tsai: *Robot Analysis and Design: The Mechanics of Serial and Parallel Manipulators* (Wiley, New York 1999)
- 2.35 K. Yamane: *Simulating and Generating Motions of Human Figures* (Springer, Berlin 2004)
- 2.36 F.M.L. Amirouche: *Fundamentals of Multibody Dynamics: Theory and Applications* (Birkhäuser, Boston 2006)
- 2.37 M.G. Coutinho: *Dynamic Simulations of Multibody Systems* (Springer, New York 2001)
- 2.38 E.J. Haug: *Computer Aided Kinematics and Dynamics of Mechanical Systems* (Allyn and Bacon, Boston, MA 1989)
- 2.39 R.L. Huston: *Multibody Dynamics* (Butterworths, Boston 1990)
- 2.40 A.A. Shabana: *Computational Dynamics*, 2nd edn. (Wiley, New York 2001)
- 2.41 V. Stejskal, M. Valášek: *Kinematics and Dynamics of Machinery* (Marcel Dekker, New York 1996)
- 2.42 L. Brand: *Vector and Tensor Analysis*, 4th edn. (Wiley/Chapman and Hall, New York/London 1953)
- 2.43 F.C. Park, J.E. Bobrow, S.R. Ploen: A Lie Group Formulation of Robot Dynamics, *Int. J. Robot. Res.* **14**(6), 609–618 (1995)
- 2.44 M.E. Kahn, B. Roth: The Near Minimum-time Control of Open-loop Articulated Kinematic Chains, *J. Dyn. Syst. Measur. Control* **93**, 164–172 (1971)
- 2.45 J.J. Uicker: Dynamic Force Analysis of Spatial Linkages, *Trans. ASME J. Appl. Mech.* **34**, 418–424 (1967)

- 2.46 A. Jain: Unified Formulation of Dynamics for Serial Rigid Multibody Systems, *J. Guid. Control Dyn.* **14**(3), 531–542 (1991)
- 2.47 G. Rodriguez: Kalman Filtering, Smoothing, and Recursive Robot Arm Forward and Inverse Dynamics, *IEEE J. Robot. Autom.* **RA-3**(6), 624–639 (1987)
- 2.48 G. Rodriguez, A. Jain, K. Kreutz-Delgado: A Spatial Operator Algebra for Manipulator Modelling and Control, *Int. J. Robot. Res.* **10**(4), 371–381 (1991)
- 2.49 J.M. Hollerbach: A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity, *IEEE Trans. Syst. Man Cybern.* **SMC-10**(11), 730–736 (1980)
- 2.50 M.W. Spong, S. Hutchinson, M. Vidyasagar: *Robot Modeling and Control* (Wiley, Hoboken, NJ 2006)
- 2.51 K.W. Buffinton: Kane's Method in Robotics. In: *Robotics and Automation Handbook*, ed. by T.R. Kurfess (CRC, Boca Raton, FL 2005), 6–1 to 6–31
- 2.52 T.R. Kane, D.A. Levinson: The Use of Kane's Dynamical Equations in Robotics, *Int. J. Robot. Res.* **2**(3), 3–21 (1983)
- 2.53 C.A. Balafoutis, R.V. Patel, P. Misra: Efficient Modeling and Computation of Manipulator Dynamics Using Orthogonal Cartesian Tensors, *IEEE J. Robot. Autom.* **4**, 665–676 (1988)
- 2.54 X. He, A.A. Goldenberg: An Algorithm for Efficient Computation of Dynamics of Robotic Manipulators. In: *Proc. of Fourth International Conference on Advanced Robotics*, (Columbus, OH, 1989) pp. 175–188
- 2.55 W. Hu, D.W. Marhefka, D.E. Orin: Hybrid Kinematic and Dynamic Simulation of Running Machines, *IEEE Trans. Robot.* **21**(3), 490–497 (2005)
- 2.56 C.A. Balafoutis, R.V. Patel: Efficient Computation of Manipulator Inertia Matrices and the Direct Dynamics Problem, *IEEE Trans. Syst. Man Cybern.* **19**, 1313–1321 (1989)
- 2.57 K.W. Lilly, D.E. Orin: Alternate Formulations for the Manipulator Inertia Matrix, *Int. J. Robot. Res.* **10**, 64–74 (1991)
- 2.58 S. McMillan, D.E. Orin: Forward dynamics of multilegged vehicles using the composite rigid body method, *Proc. IEEE International Conference on Robotics and Automation* (1998) pp. 464–470
- 2.59 U.M. Ascher, D.K. Pai, B.P. Cloutier: Forward Dynamics: Elimination Methods, and Formulation Stiffness in Robot Simulation, *Int. J. Robot. Res.* **16**(6), 749–758 (1997)
- 2.60 R. Featherstone: A Divide-and-Conquer Articulated-Body Algorithm for Parallel $O(\log(n))$ Calculation of Rigid-Body Dynamics. Part 2: Trees, Loops and Accuracy, *Int. J. Robot. Res.* **18**(9), 876–892 (1999)
- 2.61 MSC Software Corporation: *Adams*, [On-line] <http://www.mscsoftware.com/> (Nov. 12 2007)
- 2.62 T. Kane, D. Levinson: *Autolev User's Manual* (Online Dynamics Inc., 2005)
- 2.63 S. McMillan, D.E. Orin, R.B. McGhee: DynaMechs: An Object Oriented Software Package for Efficient Dynamic Simulation of Underwater Robotic Vehicles. In: *Underwater Robotic Vehicles: Design and Control*, ed. by J. Yuh (TSI Press, Albuquerque, NM 1995) pp. 73–98
- 2.64 R. Smith: *Open Dynamics Engine User Guide*, Available online: <http://www.ode.org> (Nov. 12 2007)
- 2.65 Microsoft Corporation: *Robotics Studio* [On-line] <http://www.microsoft.com/robotics> (Nov. 12 2007)
- 2.66 P.I. Corke: A Robotics Toolbox for MATLAB, *IEEE Robot. Autom. Mag.* **3**(1), 24–32 (1996)
- 2.67 M.G. Hollars, D.E. Rosenthal, M.A. Sherman: *SD/FAST User's Manual* (Symbolic Dynamics Inc., 1994)
- 2.68 G.D. Wood, D.C. Kennedy: *Simulating Mechanical Systems in Simulink with SimMechanics* (MathWorks Inc., 2003)
- 2.69 Cyberbotics Ltd.: *Webots User Guide*, Available online: <http://www.cyberbotics.com> (Nov. 8 2007)
- 2.70 I.C. Brown, P.J. Larcombe: A Survey of Customised Computer Algebra Programs for Multibody Dynamic Modelling. In: *The Use of Symbolic Methods in Control System Analysis and Design*, ed. by N. Munro (The Institute of Engineering and Technology, London 1999) pp. 53–77
- 2.71 J.J. Murray, C.P. Neuman: ARM: An algebraic robot dynamic modeling program. In: *Proc. of IEEE International Conference on Robotics and Automation*, Atlanta, Georgia, March (1984) pp. 103–114
- 2.72 J.J. Murray, C.P. Neuman: Organizing Customized Robot Dynamic Algorithms for Efficient Numerical Evaluation, *IEEE Trans. Syst. Man Cybern.* **18**(1), 115–125 (1988)
- 2.73 F.C. Park, J. Choi, S.R. Ploen: Symbolic Formulation of Closed Chain Dynamics in Independent Coordinates, *Mech. Machine Theory* **34**, 731–751 (1999)
- 2.74 M. Vukobratovic, N. Kircanski: Real-time Dynamics of Manipulation Robots. In: *Scientific Fundamentals of Robotics*, Vol. 4 (Springer-Verlag, New York 1985)
- 2.75 J. Wittenburg, U. Wolz: Mesa Verde: A Symbolic Program for Nonlinear Articulated-Rigid-Body Dynamics. In: *ASME Design Engineering Division Conference and Exhibit on Mechanical Vibration and Noise*, Cincinnati, Ohio, ASME Paper No. 85-DET-151, 1–8, September (1985)
- 2.76 J.Y.S. Luh, C.S. Lin: Scheduling of Parallel Computation for a Computer-Controlled Mechanical Manipulator, *IEEE Trans. Syst. Man Cybern.* **12**(2), 214–234 (1982)
- 2.77 D.E. Orin: Pipelined Approach to Inverse Plant Plus Jacobian Control of Robot Manipulators. In: *Proc. of IEEE International Conference on Robotics and Automation*, Atlanta, Georgia, 169–175, March (1984)
- 2.78 R.H. Lathrop: Parallelism in Manipulator Dynamics, *Int. J. Robot. Res.* **4**(2), 80–102 (1985)
- 2.79 C.S.G. Lee, P.R. Chang: Efficient Parallel Algorithm for Robot Inverse Dynamics Computation, *IEEE Trans. Syst. Man Cybern.* **16**(4), 532–542 (1986)
- 2.80 M. Amin-Javaheri, D.E. Orin: Systolic Architectures for the Manipulator Inertia Matrix, *IEEE Trans. Syst. Man Cybern.* **18**(6), 939–951 (1988)

- 2.81 C.S.G. Lee, P.R. Chang: Efficient Parallel Algorithms for Robot Forward Dynamics Computation, IEEE Trans. Syst. Man Cybern. **18**(2), 238–251 (1988)
- 2.82 M. Amin-Javaheri, D.E. Orin: Parallel Algorithms for Computation of the Manipulator Inertia Matrix, Int. J. Robot. Res. **10**(2), 162–170 (1991)
- 2.83 A. Fijany, A.K. Bejczy: A Class of Parallel Algorithms for Computation of the Manipulator Inertia Matrix, IEEE Trans. Robot. Autom. **5**(5), 600–615 (1989)
- 2.84 S. McMillan, P. Sadayappan, D.E. Orin: Parallel Dynamic Simulation of Multiple Manipulator Systems: Temporal Versus Spatial Methods, IEEE Trans. Syst. Man Cybern. **24**(7), 982–990 (1994)
- 2.85 A. Fijany, I. Sharf, G.M.T. D’Eleuterio: Parallel $O(\log N)$ Algorithms for Computation of Manipulator Forward Dynamics, IEEE Trans. Robot. Autom. **11**(3), 389–400 (1995)
- 2.86 R. Featherstone: A Divide-and-Conquer Articulated-Body Algorithm for Parallel $O(\log(n))$ Calculation of Rigid-Body Dynamics. Part 1: Basic Algorithm, Int. J. Robot. Res. **18**(9), 867–875 (1999)
- 2.87 R. Featherstone, A. Fijany: A Technique for Analyzing Constrained Rigid-Body Systems and Its Application to the Constraint Force Algorithm, IEEE Trans. Robot. Autom. **15**(6), 1140–1144 (1999)
- 2.88 P.S. Freeman, D.E. Orin: Efficient Dynamic Simulation of a Quadruped Using a Decoupled Tree-Structured Approach, Int. J. Robot. Res. **10**, 619–627 (1991)
- 2.89 Y. Nakamura, K. Yamane: Dynamics Computation of Structure-Varying Kinematic Chains and Its Application to Human Figures, IEEE Trans. Robot. Autom. **16**(2), 124–134 (2000)

Mechanisms

3. Mechanisms and Actuation

Victor Scheinman, J. Michael McCarthy

This chapter focuses on the principles that guide the design and construction of robotic systems. The kinematics equations and Jacobian of the robot characterize its range of motion and mechanical advantage, and guide the selection of its size and joint arrangement. The tasks a robot is to perform and the associated precision of its movement determine detailed features such as mechanical structure, transmission, and actuator selection. Here we discuss in detail both the mathematical tools and practical considerations that guide the design of mechanisms and actuation for a robot system.

The following sections discuss characteristics of the mechanisms and actuation that affect the performance of a robot. The first four sections discuss the basic features of a robot manipulator and their relationship to the mathematical model that is used to characterize its performance. The next two sections focus on the details of the structure and actuation of the robot and how they combine to yield various types of robots. The final section relates these design features to various performance metrics.

3.1	Overview	67
3.2	System Features	68
3.2.1	Work Envelope	68
3.2.2	Load Capacity	68
3.2.3	Kinematic Skeleton	69
3.3	Kinematics and Kinetics	69
3.3.1	Robot Topology	69
3.3.2	Kinematics Equations	70
3.3.3	Configuration Space	71
3.3.4	Speed Ratios	71
3.3.5	Mechanical Advantage	71
3.4	Serial Robots	72
3.4.1	Design Optimization	72
3.4.2	Speed Ratios	73
3.5	Parallel Robots	73
3.5.1	Workspace	73
3.5.2	Mechanical Advantage	74
3.5.3	Specialized Parallel Robots	74
3.6	Mechanical Structure	75
3.6.1	Links	75
3.6.2	Joints	76
3.7	Joint Mechanisms	76
3.7.1	Joint Axis Structures	76
3.7.2	Actuators	78
3.7.3	Transmissions	80
3.8	Robot Performance	82
3.8.1	Robot Speed	82
3.8.2	Robot Acceleration	83
3.8.3	Repeatability	83
3.8.4	Resolution	83
3.8.5	Accuracy	83
3.8.6	Component Life and Duty Cycle	83
3.8.7	Collisions	84
3.9	Conclusions and Further Reading	84
	References	84

3.1 Overview

The physical structure such as the beams, links, castings, shafts, slides, and bearings of a robot that create its movable skeleton is termed the mechanical structure or mechanism of the robot. The motors, hydraulic or pneumatic pistons, or other elements that cause the

links of the mechanism to move are called actuators. In this chapter we consider the variety of designs for the mechanisms and actuators that result in a machine system that transforms computer commands into versatile physical movement.

Early robots were designed with general motion capability under the assumption that they would find the largest market if they could perform the widest variety of tasks; this emphasis on flexibility proved to be expensive in both cost and performance. Robots are now beginning to be designed with a specific set of tasks in mind.

Robot design focuses on the number of joints, physical size, payload capacity, and the movement requirements of the end-effector. The configuration of the movable skeleton and the overall size of the robot are determined by task requirements for reach, workspace, and reorientation ability. These features affect the precision of end-effector path control needed for arc-welding

and for the smooth movement of paint spraying. They also define the absolute positioning capability necessary for assembly, the repeatability needed for materials handling, and the fine resolution that allows precise, real-time sensor-based motions.

A critical concern in robotic system design is the range of tasks the robot is expected to perform. The robot should be designed to have the flexibility it needs to perform the range of tasks for which it is intended. This determines the topology of the robot mechanism and the actuator system. The choices of geometry, material, sensors, and cable routing follow from these basic decisions.

3.2 System Features

The primary features that characterize a robot are its work envelope and load capacity.

3.2.1 Work Envelope

The space in which a robot can operate is its work envelope, which encloses its workspace. While the workspace of the robot defines positions and orientations that it can achieve to accomplish a task, the work envelope also includes the volume of space the robot itself occupies as it moves. This envelope is defined by the types of joints, their range of movement and the lengths of the links that connect them. The physical size of this envelope and the loads on the robot within this envelope are of primary consideration in the design of the mechanical structure of a robot.

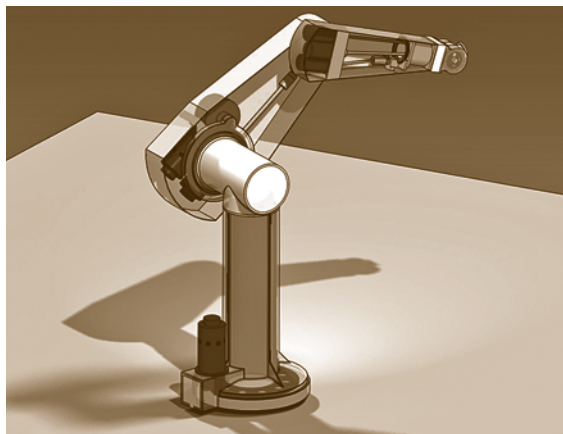


Fig. 3.1 The PUMA 560 robot

Robot work envelope layouts must include considerations of regions of limited accessibility where the mechanical structure may experience movement limitations. These constraints arise from limited joint travel range, link lengths, the angles between axes, or a combination of these. Revolute joint manipulators generally work better in the middle of their work envelopes than at extremes (Fig. 3.1). Manipulator link lengths and joint travel should be chosen to leave margins for variable sensor-guided controlled path motions and for tool or end-effector changes, as offsets and length differences will often alter the work envelope.

3.2.2 Load Capacity

Load capacity, a primary robot specification, is closely coupled with acceleration and speed. For assembly robots, mechanism acceleration and stiffness (structure and drive stiffness) are often more important design parameters than peak velocity or maximum load capacity, as minimizing small pick-and-place motion cycle time, while maintaining placement precision, is generally a top priority. In the case of arc-welding, where slow controlled-path motion is required, velocity jitter and weld path-following accuracy are important. Load capacity should be seen as a variable. It is wise to design and specify a manipulator in terms of useful payload as a function of performance rather than just in terms of maximum capacity.

Load capacity specifications must take into account gravity and inertial loading seen at the end-effector. These factors strongly affect wrist, end-effector design and drive selection. In general, load capacity is

more a function of manipulator acceleration and wrist torque than any other factor. The load rating also affects manipulator static structural deflection, steady-state motor torque, system natural frequency, damping, and the choice of servosystem control parameters for best performance and stability.

3.2.3 Kinematic Skeleton

Manipulator shape and size is determined by requirements on its workspace shape and layout, the precision of its movement, its acceleration and speed, and its construction. Cartesian manipulators (with or without revolute wrist axes) have the simplest transform and control equation solutions. Their prismatic (straight-line motion), perpendicular axes make motion planning and computation easy and relatively straightforward. Because their major motion axes do not couple dynamically (to a first order), their control equations are also simplified. Manipulators with all revolute joints are generally harder to control, but they feature a more compact and efficient structure for a given working volume. It is generally easier to design and build a good revolute joint than a long motion prismatic joint. The workspaces of revolute joint manipulators can easily overlap for coordinated multirobot installations, in contrast to gantry-style robots.

3.3 Kinematics and Kinetics

The dynamics of a robot can be separated into the properties of the movement that depend upon the geometry of its mechanical structure, termed *kinematics*, and those that depend on forces that act on the system, known as *kinetics*. It is a law of dynamics that the difference between the change in energy of the moving robot and the work performed by the forces acting on it does not change over small variations of its trajectory. This is called the *principle of virtual work* and states that variations in work and energy must cancel for all virtual displacements [3.1,2].

Because machines such as robots are designed to minimize energy losses, often due to joint friction and material strain losses, we can assume that the variation in energy is small. This means that the work input of the actuators is nearly equal to the work of the output forces.

If we consider this relationship over a small duration of time, we have that the time rate of input work, or input

Final selection of the robot configuration should capitalize on specific kinematic, structural or performance requirements. For example, a requirement for a very precise vertical straight-line motion may dictate the choice of a simple prismatic vertical axis rather than two or three revolute joints requiring coordinated control.

Six degrees of freedom (DOFs) are the minimum required to place the end-effector or tool of a robotic manipulator at any arbitrary location (position and orientation) within its accessible workspace. Most simple or preplanned tasks can be performed with fewer than six DOFs. This is because they can be carefully set up to eliminate certain axis motions, or because the tool or task does not require full specification of location. An example of this is vertical assembly using a powered screwdriver, where all operations can be achieved with three degrees of freedom.

Some applications require the use of manipulators with more than six DOFs, in particular when mobility or obstacle avoidance are necessary. For example, a pipe-crawling maintenance robot requires control of the robot's shape as well as precise positioning of its end-effector. Generally, adding degrees of freedom increases cycle time and reduces load capacity and accuracy for a given manipulator configuration and drive system.

power is nearly equal to the associated output power. Because power is force times velocity, we obtain the fundamental relationship that the ratio of input to output forces is the reciprocal of the ratio of input to output speeds. Another way of saying this is that, in the ideal machine, *the mechanical advantage of a machine is the inverse of its speed ratio*.

3.3.1 Robot Topology

The kinematic skeleton of a robot is modeled as a series of links connected by either hinged or sliding joints forming a serial chain. This skeleton has two basic forms, that of a single serial chain called a *serial robot* (Fig. 3.1) and as a set of serial chains supporting a single end-effector, called a *parallel robot*, such as the platform shown in Fig. 3.2. Robots can be configured to work in parallel such as the individual legs of walking machines (Figs. 3.3 and 3.4) [3.3], as

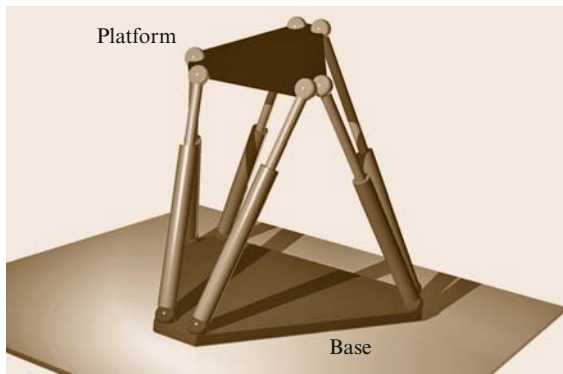


Fig. 3.2 A parallel robot can have as many as six serial chains that connect a platform to the base frame



Fig. 3.3 A photograph of the adaptive suspension vehicle (ASV) walking machine

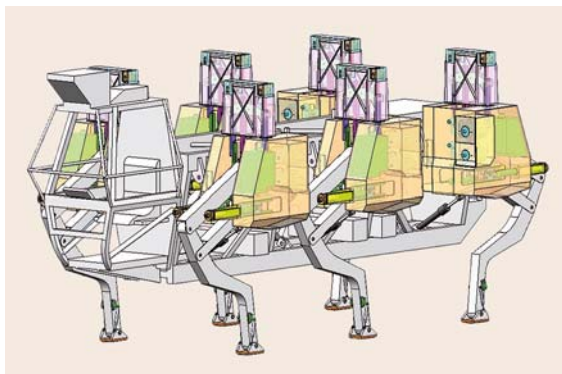


Fig. 3.4 The adaptive suspension vehicle walking machine

well as the fingers of mechanical hands (Figs. 3.6 and 3.7) [3.4].

The robot end-effector is the preferred tool for interaction with the environment, and the ability to position and orient this end-effector is defined by the skeleton of

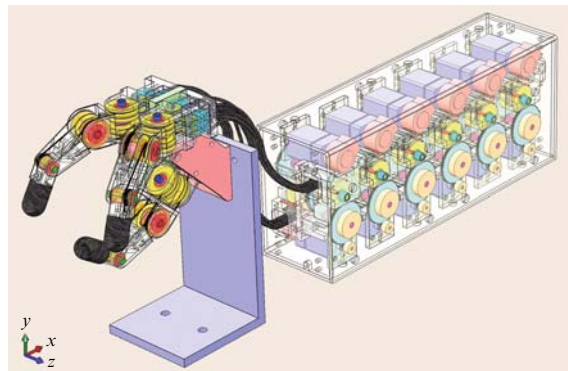


Fig. 3.5 The Salisbury three-fingered robot hand with its cable drive system

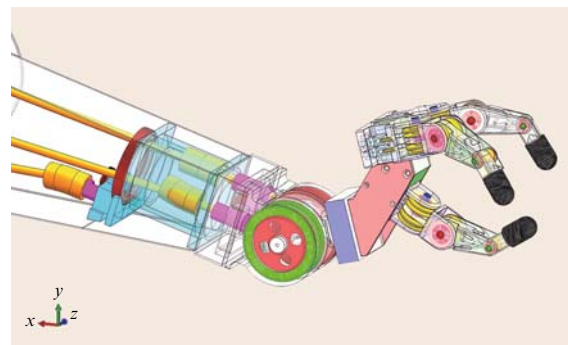


Fig. 3.6 The Salisbury hand as the end-effector of a PUMA robot (the drive system is not shown)

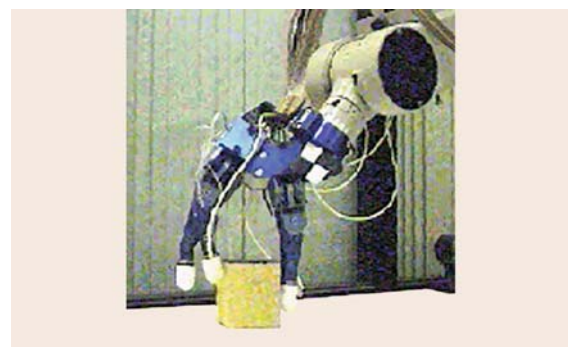


Fig. 3.7 A photograph of the Salisbury three-fingered hand grasping a block

the robot. In a general serial robot, a chain of six joints provides full control over the end-effector. In a general parallel robot, there are more than six joints and the six actuators may be applied to these joints in a variety of ways to control the movement of the end-effector.

3.3.2 Kinematics Equations

A robot is designed so that specifying the values of local joint parameters, such as the angle of rotary joints and the travel of sliding joints, specifies the position of every component of a machine using its kinematics equations. To do this the robot is described by a sequence of lines representing the axes \hat{z}_j of equivalent revolute or prismatic joints and the common normal lines \hat{x}_j which form the kinematic skeleton of the chain (Fig. 3.2). This construction allows the specification of the location of each link of the robot relative to the base by the matrix equation

$$\mathbf{T} = \mathbf{Z}(\theta_1, d_1)\mathbf{X}(\alpha_1, a_1)\mathbf{Z}(\theta_2, d_2) \dots \times \mathbf{X}(\alpha_{m-1}, a_{m-1})\mathbf{Z}(\theta_m, d_m), \quad (3.1)$$

known as the *kinematics equations* of the chain [3.5, 6] (see Chap. 1; (1.44)). The set of all positions \mathbf{T} that the end-effector can reach for all values of the joint parameters is called the *workspace* of the robot.

The matrices $\mathbf{Z}(\theta_j, d_j)$ and $\mathbf{X}(\alpha_j, a_j)$ are 4×4 matrices that define screw displacements around and along the joint axes \hat{z}_j and \hat{x}_j , respectively [3.7]. The parameters α_j and a_j define the dimensions of the links in the chain. The parameter θ_j is the joint variable for revolute joints and d_j is the variable for prismatic joints. The trajectory ${}^F\mathbf{p}(t)$ of a point ${}^M\mathbf{p}$ in the end-effector is obtained from the joint trajectory, $\mathbf{q}(t) = (q_1(t), \dots, q_m(t))^T$, where q_i is either θ_i or d_i depending on the joint, given by

$${}^F\mathbf{p}(t) = \mathbf{T}(\mathbf{q}(t)){}^M\mathbf{p}. \quad (3.2)$$

If the end-effector is connected to the base frame by more than one serial chain (Fig. 3.2) then we have a set of kinematics equations for each chain,

$$\mathbf{T} = \mathbf{B}_j\mathbf{T}(\mathbf{q}_j)\mathbf{E}_j, \quad j = 1, \dots, n, \quad (3.3)$$

where \mathbf{B}_j locates the base of the j -th chain and \mathbf{E}_j defines the position of its attachment to the part, or end-effector. The set of positions \mathbf{T} that simultaneously satisfy all of these equations is the workspace of the part. This imposes constraints on the joint variables that must be determined to define its workspace completely [3.8, 9].

3.3.3 Configuration Space

The kinematics equations of the robot relate the range of values available for the joint parameters, called the configuration space of the robot, to the workspace of the end-effector. This configuration space is a fundamental

tool in robot path planning for obstacle avoidance [3.10]. Though any link in the chain forming a robot may hit an obstacle, it is the end-effector that is intended to approach and move around obstacles such as the table supporting the robot and the fixtures for parts it is to pick up. Obstacles define forbidden positions and orientations in the workspace which map back to forbidden joint angles in the configuration space of the robot. Robot path planners seek trajectories to a goal position through the free space around these joint space obstacles [3.11].

3.3.4 Speed Ratios

The speed ratios of a robot relate the velocity ${}^F\dot{\mathbf{p}}$ of a point ${}^F\mathbf{p}$ in the end-effector to the joint rates $\dot{\mathbf{q}} = (\dot{q}_1, \dots, \dot{q}_m)^T$, that is

$${}^F\dot{\mathbf{p}} = \mathbf{v} + \boldsymbol{\omega} \times ({}^F\mathbf{p} - \mathbf{d}), \quad (3.4)$$

where \mathbf{d} and \mathbf{v} are the position and velocity of a reference point, respectively, and $\boldsymbol{\omega}$ is the angular velocity of the end-effector.

The vectors \mathbf{v} and $\boldsymbol{\omega}$ depend on the joint rates \dot{q}_j through the formula

$$\begin{pmatrix} \mathbf{v} \\ \boldsymbol{\omega} \end{pmatrix} = \begin{pmatrix} \frac{\partial \mathbf{v}}{\partial \dot{q}_1} & \frac{\partial \mathbf{v}}{\partial \dot{q}_2} & \dots & \frac{\partial \mathbf{v}}{\partial \dot{q}_m} \\ \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_1} & \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_2} & \dots & \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_m} \end{pmatrix} \begin{pmatrix} \dot{q}_1 \\ \vdots \\ \dot{q}_m \end{pmatrix}, \quad (3.5)$$

or

$$\mathbf{v} = \mathbf{J}\dot{\mathbf{q}}. \quad (3.6)$$

The coefficient matrix \mathbf{J} in this equation is called the *Jacobian* and is a matrix of speed ratios relating the velocity of the tool to the input joint rotation rates [3.6, 9].

3.3.5 Mechanical Advantage

If the end-effector of the robot exerts a force \mathbf{f} at the point ${}^F\mathbf{p}$, then the power output is

$$P_{\text{out}} = \mathbf{f} \cdot {}^F\dot{\mathbf{p}} = \sum_{j=1}^m \mathbf{f} \cdot \left[\frac{\partial \mathbf{v}}{\partial \dot{q}_j} + \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_j} \times ({}^F\mathbf{p} - \mathbf{d}) \right] \dot{q}_j. \quad (3.7)$$

Each term in this sum is the portion of the output power that can be associated with an actuator at joint \mathbf{S}_j , if one exists.

The power input at joint \mathbf{S}_j is the product $\tau_j \dot{q}_j$ of the torque τ_j and joint angular velocity \dot{q}_j . Using the principle of virtual work for each joint we can compute

$$\tau_j = \mathbf{f} \cdot \frac{\partial \mathbf{v}}{\partial \dot{q}_j} + ({}^F\mathbf{p} - \mathbf{d}) \times \mathbf{f} \cdot \frac{\partial \boldsymbol{\omega}}{\partial \dot{q}_j}, \quad j = 1, \dots, m. \quad (3.8)$$

We have arranged this equation to introduce the force-torque vector $\mathbf{f} = (\mathbf{f}, (\mathbf{p} - \mathbf{d}) \times \mathbf{f})^\top$ at the reference point \mathbf{d} .

The equations (3.8) can be assembled into the matrix equation

$$\boldsymbol{\tau} = \mathbf{J}^\top \mathbf{f} \quad (3.9)$$

3.4 Serial Robots

A serial chain robot is a sequence of links and joints that begins at a base and ends with an end-effector (Fig. 3.8). The links and joints of a robot are often configured to provide separate translation and orientation structures. Usually, the first three joints are used to position a reference point in space and the last three form the *wrist* which orients the end-effector around this point [3.12, 13]. This reference point is called the *wrist center*. The volume of space in which the wrist center can be placed is called the *reachable workspace* of the robot. The rotations available at each of these points is called the *dexterous workspace*.

The design of a robot is often based on the symmetry of its reachable workspace. From this point of view there are three basic shapes: rectangular, cylindrical, and spherical [3.6]. A rectangular workspace is provided by three mutually perpendicular prismatic (P) joints which form a PPPS chain called a *Cartesian* robot – S denotes a spherical wrist which allows all rotations about its center point. A rotary base joint combined with two prismatic joints forms a CPS chain with a cylindrical workspace – C denotes a rotary (R) and sliding (P) joint with the same axis. The P-joint can be replaced by a revolute (R) joint that acts as an elbow in order to provide the same radial movement. Finally, two rotary joints at right angles form a T-joint at the base of the robot that

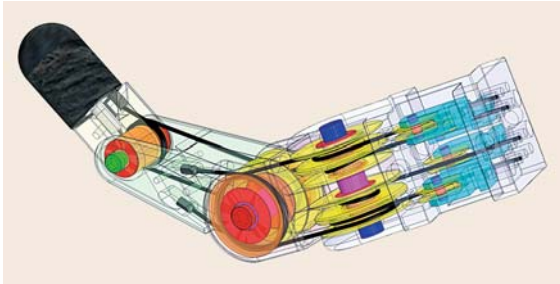


Fig. 3.8 A single finger of the Salisbury hand is a serial chain robot

where \mathbf{J} is the *Jacobian* defined above in (3.5). For a chain with six joints this equation can be solved for the output force-torque vector \mathbf{f} ,

$$\mathbf{f} = (\mathbf{J}^\top)^{-1} \boldsymbol{\tau} \quad (3.10)$$

Thus, the matrix that defines the mechanical advantage for this system is the inverse of the matrix of speed ratios.

supports rotations about a vertical and horizontal axes. Radial movement is provided either by a P-joint, or by an R-joint configured as an elbow. The result is a TPS or TRS chain with a spherical workspace.

It is rare that the workspace is completely symmetrical because joint axes are often offset to avoid link collisions and there are limits to joint travel which combine to distort the shape of the workspace.

3.4.1 Design Optimization

Another approach to robot design uses a direct specification of the workspace as a set of positions for the end-effector of a robotic system [3.14–17] which we call the *task space*. A general serial robot arm has two design parameters, link offset and twist, for each of five links combined with four parameters each that locate the base of the robot and the workpiece in its end-effector, making a total of 18 design variables. The link parameters are often specified so the chain has a spherical wrist and specific workspace shape. The design goal is usually to determine the workspace volume and locate the base and workpiece frames so that the workspace encloses the specified task space.

The task space is defined by a set of 4×4 transformations \mathbf{D}_i , $i = 1, \dots, k$. The problem is solved iteratively by selecting a design and using the associated kinematics equations $\mathbf{T}(\mathbf{q})$ to evaluate relative displacements in the objective function

$$f(\mathbf{r}) = \sum_{i=1}^k \|\mathbf{D}_i \mathbf{T}^{-1}(\mathbf{q}_i)\| \quad (3.11)$$

Optimization techniques yield the design parameter vector \mathbf{r} that minimizes this objective function.

This optimization depends on the definition of the distance measure between the positions reached by the end-effector and the desired workspace. Park [3.18], Martinez and Duffy [3.19], Zefran et al. [3.20], Lin and

Burdick [3.21], and others have shown that there is no distance metric that is coordinate frame invariant. This means that, unless this objective function can be forced to zero so the workspace completely contains the task space, the resulting design will not be *geometric* in the sense that the design is not independent of the choice of coordinates.

3.4.2 Speed Ratios

A six-axis robot has a 6×6 Jacobian \mathbf{J} obtained from (3.5) that is an array of speed ratios relating the components of the velocity \mathbf{v} of the wrist center and the angular velocity $\boldsymbol{\omega}$ of the end-effector to each of the joint velocities. Equation (3.9) shows that this Jacobian defines the force-torque vector \mathbf{f} exerted at the wrist center in terms of the torque applied by each of the actuators. The link parameters of the robot can be selected to provide a Jacobian \mathbf{J} with specific properties.

The sum of the squares of the actuator torques of a robot is often used as a measure of *effort* [3.22, 23]. From (3.9) we have

$$\boldsymbol{\tau}^\top \boldsymbol{\tau} = \mathbf{f}^\top \mathbf{J} \mathbf{J}^\top \mathbf{f} . \quad (3.12)$$

The matrix $\mathbf{J} \mathbf{J}^\top$ is square and positive definite. Therefore, it can be viewed as defining a hyperellipsoid in

six-dimensional space [3.24]. The lengths of the semi-diameters of this ellipsoid are the inverse of the absolute value of the eigenvalues of the Jacobian \mathbf{J} . These eigenvalues may be viewed as *modal* speed ratios that define the amplification associated with each joint velocity. Their reciprocals are the associated *modal* mechanical advantages, so the shape of this ellipsoid illustrates the force amplification properties of the robot.

The ratio of the largest of these eigenvalues to the smallest, called the condition number, gives a measure of the anisotropy or *out-of-roundness* of the ellipsoid. A sphere has a condition number of one and is termed *isotropic*. When the end-effector of a robot is in a position with an isotropic Jacobian there is no amplification of the speed ratios or mechanical advantage. This is considered to provide high-fidelity coupling between the input and output because errors are not amplified [3.25, 26]. Thus, the condition number is used as a criterion in a robot design [3.27].

In this case, it is assumed that the basic design of the robot provides a workspace that includes the task space. Parameter optimization finds the internal link parameters that yield the desired properties for the Jacobian. As in minimizing the distance to a desired workspace, optimization based on the Jacobian depends on a careful formulation to avoid coordinate dependency.

3.5 Parallel Robots

A robotic system in which two or more serial chain robots support an end-effector is called a *parallel robot*. For example, the adaptive suspension vehicle (ASV) leg (Fig. 3.9), is a pantograph mechanism driven by parallel actuation. Each supporting chain of a parallel robot may have as many as six degrees of freedom, however, in general only a total of six joints in the entire system are actuated. A good example is the Stewart platform formed from six TPS robots in which usually only the prismatic joint (P-joint) in each chain is actuated (Fig. 3.2) [3.9, 28, 29].

The kinematics equations of the TPS legs are

$$\mathbf{T} = \mathbf{B}_j \mathbf{T}(\theta_j) \mathbf{E}_j, \quad j = 1, \dots, 6, \quad (3.13)$$

where \mathbf{B}_j locates the base of the leg and \mathbf{E}_j defines the position of its attachment to the end-effector. The set of positions \mathbf{T} that simultaneously satisfy all of these equations is the workspace of the parallel robot.

Often the workspace of an individual chain of a parallel robot can be defined by geometric constraints, for

example, a position \mathbf{T} is in the workspace of the j -th supporting TPS chain if it satisfies the constraint equation

$$(\mathbf{T} \mathbf{x}_j - \mathbf{p}_j) \cdot (\mathbf{T} \mathbf{x}_j - \mathbf{p}_j) = \rho_j^2 . \quad (3.14)$$

This equation defines the distance between the base joint \mathbf{p}_j and the point of attachment ${}^E \mathbf{x}_j = \mathbf{T} \mathbf{x}_j$ to the platform as the length ρ_j is controlled by the actuated prismatic joint. In this case the workspace is the set of positions \mathbf{T} that satisfy all six equations, one for each leg.

3.5.1 Workspace

The workspace of a parallel robot is the intersection of the workspaces of the individual supporting chains. However, it is not the intersection of the reachable and dexterous workspaces separately. These workspaces are intimately combined in parallel robots. The dexterous workspace is usually largest near the center of the reachable workspace and shrinks as the reference point moves

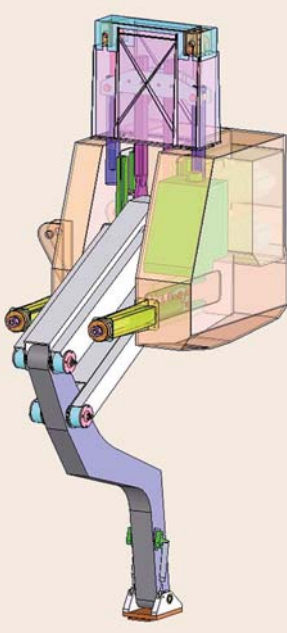


Fig. 3.9 One leg of the ASV walking machine is a parallel robot

toward the edge. A focus on the symmetry of movement allowed by supporting leg designs has been an important design tool resulting in many novel parallel designs [3.30, 31]. Simulation of the system is used to evaluate its workspace in terms of design parameters.

Another approach is to specify directly the positions and orientations that are to lie in the workspace and solve the algebraic equations that define the leg constraints to determine the design parameters [3.32, 33]. This is known as kinematic synthesis and yields parallel robots that are asymmetric but have specified reachable and dexterous workspaces, see McCarthy [3.34].

3.5.2 Mechanical Advantage

The force amplification properties of a parallel robot are obtained by considering the Jacobians of the individual supporting chains. Let the linear and angular velocity of the platform be defined by the six-vector $\mathbf{v} = (\mathbf{v}, \boldsymbol{\omega})^\top$, then from the kinematics equations of each of the support legs we have

$$\mathbf{v} = \mathbf{J}_1 \dot{\rho}_1 = \mathbf{J}_2 \dot{\rho}_2 = \cdots = \mathbf{J}_6 \dot{\rho}_6. \quad (3.15)$$

Here we assume that the platform is supported by six chains, but it can be fewer, such as when the fingers of a mechanical hand grasp an object [3.4].

The force on the platform applied by each chain is obtained from the principle of virtual work as

$$\mathbf{f}_j = (\mathbf{J}_j^\top)^{-1} \boldsymbol{\tau}_j, \quad j = 1, \dots, 6. \quad (3.16)$$

There are only six actuated joints in the system so we assemble the associated joint torques into the vector $\boldsymbol{\tau} = (\tau_1, \dots, \tau_6)^\top$. If \mathbf{f}_i is the force-torque vector obtained from (3.16) for $\tau_i = 1$ and the remaining torques to zero, then the resultant force-torque \mathbf{w} applied to the platform is

$$\mathbf{w} = (\mathbf{f}_1, \mathbf{f}_2, \dots, \mathbf{f}_6) \boldsymbol{\tau}, \quad (3.17)$$

or

$$\mathbf{w} = \boldsymbol{\Gamma} \boldsymbol{\tau}. \quad (3.18)$$

The elements of the coefficient matrix $\boldsymbol{\Gamma}$ define the mechanical advantage for each of the actuated joints. In the case of a Stewart platform the columns of this matrix are the Plücker coordinates of the lines along each leg [3.29].

The principle of virtual work yields the velocity of the platform in terms of the joints rates $\dot{\rho}$ as

$$\boldsymbol{\Gamma}^\top \mathbf{v} = \dot{\rho}. \quad (3.19)$$

Thus, the inverse of $\boldsymbol{\Gamma}$ defines the speed ratios between the actuated joints and the end-effector. The same equation can be obtained by computing the derivative of the geometric constraint equations (3.14), and $\boldsymbol{\Gamma}$ is the *Jacobian* of the parallel robot system [3.35].

The Jacobian $\boldsymbol{\Gamma}$ is used in parameter optimization algorithms to design parallel robots [3.36] with isotropic mechanical advantage. The square root of the determinant $|\boldsymbol{\Gamma} \boldsymbol{\Gamma}^\top|$ measures the six-dimensional volume spanned by the column vectors \mathbf{f}_j . The distribution of the percentage of this volume compared to its maximum within the workspace is also used as a measure of the overall performance [3.37, 38]. A similar performance measure normalizes this Jacobian by the maximum joint torques available and the maximum component of force and torque desired, and then seeks an isotropic design [3.39].

3.5.3 Specialized Parallel Robots

Another approach to the design of parallel robots has been to separate their functionality into orientation and translational platforms. Tsai and Joshi [3.40] and Jin and Yang [3.41] survey designs for a class of parallel chains that generate pure translation. Kong and Gosselin [3.42] and Hess-Coelho [3.43] do the same for parallel chains that provide rotational movement in space.

3.6 Mechanical Structure

For the purposes of dynamic modeling, the links of a robot are generally considered to be rigid. However, a robot is not a rigid structure. Like all structures it deflects under applied loads, such as its own weight and the weight of the payload, termed gravity loading; see Figs. 3.10 and 3.11. The issue is a matter of degree. The more force that is needed to cause a deflection in the links, the more the robot moves like a connected set of rigid bodies. Rigid robots have links designed to be stiff so the deflections under load are less than the positioning accuracy required for their range of tasks. This allows the dynamic model and control algorithms to ignore link deflection. Most commercially available robot arms are of this type (see Rivin [3.44]).

It is possible to improve the positioning accuracy of a rigid robot by augmenting a control algorithm that includes a model of link deflection resulting from gravity loading. It is also possible to use strain sensors to measure loads and deflections. These *semirigid* robots assume small structural deflections that are linearly related to known applied loads.

Flexible robots require that the dynamic model include the deflection of its various links under gravity loading as well as under the forces associated with link acceleration, called inertia loading. The robot control algorithms must control the vibration of the system as well its gross motion. Management of vibration is required even in rigid robots to achieve high speed and manipulate large payloads.

3.6.1 Links

For industrial robots a critical concern is the link stiffness in bending and in torsion. To provide this stiffness, robot links are designed either as beams or shell (monocoque) structures. Monocoque structures have lower weight or higher strength-to-weight ratios, but are more costly and generally more difficult to manufacture. Cast, extruded, or machined beam-based links are often more cost effective; see Juvinall and Marshek [3.45], and Sigley and Mischke [3.46, 47]

Another important consideration is whether the link structure includes bolted, welded, or adhesive bonded assemblies of cast, machined, and fabricated elements. Screw and bolted connections may seem straightforward, inexpensive, and easily maintained, but the inevitable deflection of a link even in the manufacturing process introduces creep in these multiple element assemblies that changes the dimensions and performance

of the robot. Welded and cast structures are much less susceptible to creep and the associated hysteresis deformation, though in many cases they require secondary manufacturing operations such as thermal stress relieving and finish machining.

The minimum practical wall or web thickness for castings may be thicker than necessary for stiffness. Thin walls can be achieved with structural skin (monocoque) structures but this is offset by the potential for denting, permanent deformation, and damage in the event of slight collisions. Therefore, the performance requirements must be considered when selecting the construction and fabrication details of the robot.



Fig. 3.10 The hydraulic Skywash aircraft cleaning robot



Fig. 3.11 The DeLaval VMS milking robot

Performance- and application-specific materials and geometry are used to reduce the weight of the links and therefore the associated gravity and inertial loading. For structures that move in a straight line, aluminum or magnesium alloy extrusions of constant cross section are convenient. Carbon and glass-fiber composites provide lower mass for robots that require high acceleration (painting robots). Thermoplastic materials provide low-cost link structures though at reduced load capacity. Stainless steel is often used in robots for medical and food service applications. Because rotary joints generate linear accelerations that increase with the distance from the axis, links attached to these joints are often designed to taper in cross section or wall thickness to reduce the associated inertial loading.

3.7 Joint Mechanisms

A robot joint mechanism consists of at least four major components: the joint axis structure, an actuator, transmission, and state sensor (usually for position feedback, but velocity and force sensors are also common).

For low-performance manipulators that accelerate the payload at less than a peak of 0.5 g, system inertia is not as important as gravity forces and torques. This means the actuators can be placed near the joints, and their suspended weight compensated using counterbalancing masses, springs, or gas pressure.

In high-performance robots where peak payload accelerations reach 3–10 g or more, minimizing system inertia is important. The actuators are placed near the first joint axis of a serial link manipulator to minimize its inertial contribution, and drive links, belts or cables or gear transmissions are used to drive the joints.

While a longer transmission distance can reduce mass and gravity moments and inertia, it introduces flexibility and thus reduces the system stiffness. The design of the actuator placement and transmission for each joint is a tradeoff between weight, inertia, stiffness, and complexity. This choice dictates the major physical characteristics of a manipulator design. To illustrate this point, consider the Adept 1 assembly robot which has four degrees of freedom, each with a different structure. The first axis has a direct motor drive. The second axis is driven by a band drive, and the third by a belt drive. Finally, the fourth axis uses a linear ball–screw drive. For a variety of useful joint mechanisms see *Sclater and Chironis* [3.48].

3.6.2 Joints

Joints for most robots allow either rotary or linear movement, termed revolute and prismatic joints. Other joints that are available are the ball-in-socket, or spherical joint, and the Hooke-type universal joint.

Integration of the mechanical structure of the robot with its joint mechanism, which includes the actuator and joint motion sensor, is a source of structural flexibility. Deformation in the joint at the bearing housings can reduce shaft and gear preloads, allowing backlash or free play, which reduces precision. Structural flexibility can also introduce changes in gear center spacing, introducing forces and torques and associated deflection, binding, jamming, and wear.

3.7.1 Joint Axis Structures

Revolute Joints

Revolute or rotary motion joints are designed to perform pure rotation while minimizing other displacements and motions. The most important measure of the quality of a revolute joint is its stiffness or resistance to all undesired motion. Key factors to be considered in design for stiffness are shaft diameter, clearances and tolerances, mounting configuration of the bearings, and the implementation of proper bearing preloading. Shaft diameter and bearing size are not always based on load-carrying capacity; rather, they often will be selected to be compatible with a rigid mounting configuration and also have a bore large enough to pass cables, hoses, and even drive elements for other joints. Because joint shafts will frequently be torque-transmitting members, they and their supporting structure must be designed both for bending and torsional stiffness. The first axis of the PUMA robot is an example of such a joint with its large-diameter tubular configuration.

An important factor in maintaining stiffness in a revolute joint is the choice of bearing-mounting configuration. The mounting arrangement and mount must be designed to accommodate manufacturing tolerances, thermal expansion and bearing preload. Axial preloading of ball or tapered roller bearings improves system accuracy and stiffness by minimizing bearing radial and axial play. Preloads can be achieved through selective assembly or elastic (spring) elements, shim spacers,

threaded collars, four-point contact bearings, duplex bearing arrangements, and tight manufacturing tolerances.

Prismatic Joints

There are two basic types of prismatic or linear motion joints: single-stage and telescoping joints. Single-stage joints are made up of a moving surface that slides linearly along a fixed surface. Telescoping joints are essentially sets of nested or stacked single-stage joints. Single-stage joints feature simplicity and high stiffness, whereas the primary advantage of telescoping joints is their retracted-state compactness and large extension ratio. Telescoping joints have a lower effective joint inertia for some motions because part of the joint may remain stationary or move with reduced acceleration.

The primary functions of bearings in prismatic joints are to facilitate motion in a single direction and to prevent motion in all other directions. Preventing these unwanted motions poses the more challenging design problem. Deformations in the structure can significantly affect bearing surface configuration, which affects performance. In severe cases, roller deflection under load may cause binding, which precludes motion. For high-precision prismatic joints, ways must be made straight over long distances. The required precision grinding on multiple surfaces can be expensive. Expensive and bulky covers are required to shield and seal a prismatic bearing and way.

The primary criterion for evaluating higher number (in or near the wrist or end-effector) linear motion joints or axes is the stiffness-to-weight ratio. Achieving a good stiffness-to-weight ratio requires the use of a hollow or thin-walled structure rather than solid members for the moving elements.

Bearing spacing is extremely important in design for stiffness. If this spacing is too short, system stiffness will be inadequate no matter how great the bearing stiffness. Major causes for failure in prismatic joints are foreign particle contamination and surface fatigue wear (brinelling) of the ways caused by excessive ball loading due to high preload, moment loads and shock loads.

The large exposed precision surfaces in most prismatic joints make them much more sensitive than revolute joints to improper handling and environmental effects. They are also significantly more difficult to manufacture, properly assemble, and align.

Common types of sliding elements for prismatic motion are bronze or thermoplastic impregnated bushings. These bushings have the advantage of being low in cost,

of having relatively high load capacity, and of working with unhardened or superficially hardened (i.e., plated or anodized) surfaces. Because the local or contact stress on the moving element is distributed and is low this element may be made of thin tubing. Another type of bushing in common use is the ball bushing. Ball bushings have the advantages of lower friction and greater precision than plain bushings. However, they require that the contacting surface of the joint be heat treated or hardened (generally to Rc 55 or greater) and of sufficient case and wall thickness to support the ball point contact loads and resulting high stresses.

Ball and roller slides are also commonly used in robot prismatic joints. There are two basic categories of these slides; recirculating and non-recirculating. Non-recirculating ball and roller slides are used primarily for short-travel applications. They feature high precision and very low friction at the expense of being more sensitive to shock and relatively poor at accommodating moment loading. Recirculating ball slides are somewhat less precise but can carry higher loads than non-recirculating ball slides. They can also be set up to handle relatively large moment loads. Travel range can be up to several meters. Commercial recirculating ball slides and ways have greatly simplified the design and construction of linear axes, particularly in gantry and track manipulators.

Another common type of prismatic robot joint is made up of cam followers, rollers, or wheels rolling on extruded, drawn, machined, or ground surfaces. In high-load applications the surfaces must be hardened before they are finish ground. Cam followers can be purchased with eccentric mounting shafts to facilitate setup and adjustment. Elastomer rollers provide quiet, smooth operation.



Fig. 3.12 RobotWorld is an integrated workcell with multiple robot modules that move on air bearings

Two less common types of linear or prismatic joints feature flexures and air bearings. Flexure-based joints, whose motion results from elastic bending deformations of beam support elements, are used primarily for small, high-resolution, quasilinear motions. Air bearings require smooth surfaces and close control of tolerances as well as a constant supply of filtered, oil-free compressed air. Two- and three-degree-of-freedom air bearings (x , y , θ) can enable multi-axis motion with few moving parts; see Fig. 3.12.

Joint Travel

For revolute joint configurations, the shoulder and elbow joints and links determine the gross volume of the work envelope (reachable workspace) of a robot manipulator arm. The wrist joints generally determine the orientation range (dexterous workspace) about a point within this work envelope. Larger joint travel may increase the number of possible manipulator configurations that will reach a particular location (increased task space). Wrist joint travel in excess of 360° and up to 720° can be useful for situations requiring controlled-path (e.g., straight-line) motion, synchronized motion such as conveyor tracking, or sensor-modified motions. Continuous last-joint rotation is desirable in certain cases like loading or unloading a rotating machine or mating threaded parts.

Additional joints and links, sometimes in the robot but more often in the end-effector, and specialized tooling also serve to increase the task space of a robot. Continuous and controlled-path robot motion requires planning to avoid singularities (regions where two or more joints may become aligned or nearly aligned) and the resulting unstable end-effector motion in these regions. Manipulator design coordinated with a well-planned workcell layout can improve the useful task space by placing critical motions well away from singularity regions. For example, a standard three-axis robot wrist has singularities 180° apart, which can be increased to 360° by implementing a somewhat more complex reduced singularity wrist. Such a wrist is used in a sheep-shearing robot to achieve multiple, long, continuous, smooth, constant-velocity, sensor-guided passes over the contoured body of a sheep to shear its wool [3.49, 50].

3.7.2 Actuators

Actuators supply the motive power for robots. Most robot actuators are commercially available components, which are adapted or modified, as necessary, for a specific robot application. The three commonly used actuators are hydraulic, pneumatic, and electromagnetic.

Hydraulic Actuators

Hydraulic actuators, chosen as power sources for the earliest industrial robots, offer very large force capability and high power-to-weight ratios. In a hydraulic system the power is provided mechanically from an electric motor or engine driven high-pressure fluid pump, see Fig. 3.10. Actuators are most commonly linear cylinders, rotary vane actuators, and hydraulic motors. Actuator control is through a solenoid valve (on/off control) or a servovalve (proportional control), which is driven electrically from a low-power electronic control circuit. The hydraulic power supply is bulky and the cost of the proportional, fast-response servovalves are high. Leaks and maintenance issues have limited the use and application of hydraulically powered robots.

Pneumatic Actuators

Pneumatic actuators are primarily found in simple manipulators. Typically they provide uncontrolled motion between mechanical limit stops. These actuators provide good performance in point-to-point motion. They are simple to control and are low in cost. Although a few small actuators may be run with typical factory air supplies, extensive use of pneumatic-actuated robots requires the purchase and installation of a costly dedicated compressed-air source. Pneumatic actuators have low energy efficiency.

Proportional, closed-loop, servo-controlled pneumatic manipulators have been developed and successfully applied, principally in applications where safety, environmental, and application conditions discourage electric drives. An example is an early version of the DeLaval International AB Tumba, Sweden VMS (Voluntary Milking System) cow-milking robot, which used pneumatic actuators and electro-pneumatic proportional valve joint controls in a farm, milking stall, environment (Fig. 3.11).

Electromagnetic Actuators

The most common types of actuators in robots today are electromagnetic actuators.

Stepper Motors. Small, simple robots, such as bench-top adhesive dispensing robots, frequently use stepper or pulse motors of the permanent magnet (PM) hybrid type or sometimes the variable reluctance (VR) type (see Fig. 3.13). These robots use open-loop position and velocity control. They are relatively low in cost and interface easily to electronic drive circuits. Microstep control can produce 10 000 or more discrete



Fig. 3.13 The Sony robot uses open-loop permanent-magnet stepper motors



Fig. 3.14 The Adept robot uses closed-loop control and variable-reluctance motors

robot joint positions. In open-loop step mode the motors and robot motions have a significant settling time, which can be damped either mechanically or through the application of control algorithms. Power-to-weight ratios are lower for stepper motors than for other types of electric motors. Stepper motors operated with closed-loop control function similarly to direct-current (DC) or alternating-current (AC) servomotors (Fig. 3.14).

Permanent-Magnet DC Motor. The permanent-magnet, direct-current, brush-commutated motor is widely available and comes in many different types and configurations. The lowest-cost permanent-magnet motors use ceramic (ferrite) magnets. Robot toys and hobby robots often use this type of motor. Neodymium (NEO) magnet motors have the highest energy-product magnets, and in general produce the most torque and power for their size.

Ironless rotor motors, often used in small robots, typically have copper wire conductors molded into epoxy or composite cup or disk rotor structures. The advantages of these motors include low inductance, low friction, and no cogging torque. Disk armature motors have several advantages. They have short overall lengths, and because their rotors have many commutation segments, they produce a smooth output with low torque ripple. A disadvantage of ironless armature motors is that they have a low thermal capacity due to low mass and limited thermal paths to their case. As a result, when driven at high power levels they have rigid duty-cycle limitations or require forced-air cooling.

Brushless Motors. Brushless motors, also called AC servomotors or brushless DC motors, are widely used in industrial robots (see Figs. 3.15 and 3.16). They substitute magnetic or optical sensors and electronic switching circuitry for the graphite brushes and copper bar commutator, thus eliminating the friction, sparking, and wear of commutating parts. Brushless motors generally have good performance at low cost because of the decreased complexity of the motor. However, the controllers for these motors are more complex and expensive than brush-type motor controllers. The brush-



Fig. 3.15 The Baldor AC servomotor



Fig. 3.16 The Anorad brushless linear motor

less motor's passive multipole neodymium magnet rotor and wire-wound iron stator provide good heat dissipation and excellent reliability. Linear brushless motors function like unrolled rotary motors. They typically have a long, heavy, multiple magnet passive stator and a short, lightweight, electronically commutated wire-wound forcer (slider).

Other Actuators. A wide variety of other types of actuators have been applied to robots. A sampling of these include, thermal, shape-memory alloy (SMA), bimetallic, chemical, piezoelectric, magnetostrictive, electroactive polymer (EPAM), bladder, and micro-electromechanical system (MEMS) actuators (see Figs. 3.17 and 3.18). Most of these actuators have been



Fig. 3.17 The artificial muscle EPAM motor

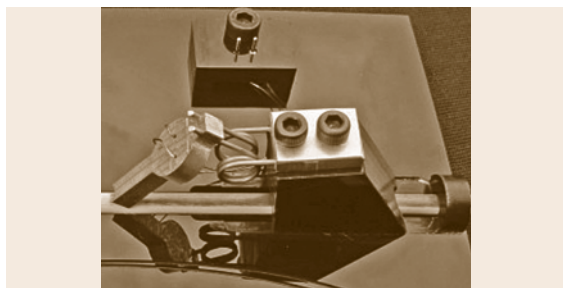


Fig. 3.18 The Elliptec piezoelectric motor



Fig. 3.19 A six-axis Physik Instrumente (PI) piezo hexapod with sub-nanometer resolution

applied to research and special application robots rather than volume production industrial robots. An example of a piezoelectric actuator powered robot is the six-axis PI piezo hexapod with sub-nanometer resolution shown in Fig. 3.19.

3.7.3 Transmissions

The purpose of a transmission or drive mechanism is to transfer mechanical power from a source to a load. The design and selection of a robot drive mechanism requires consideration of motion, load, and power requirements and the placement of the actuator with respect to the joint. The primary considerations in transmission design are stiffness, efficiency, and cost. Backlash and windup impact drive stiffness especially in robot applications where motion is constantly reversing and loading is highly variable. High transmission stiffness and low or no backlash result in increased friction losses. Most robot transmission elements have good efficiencies when they are operating at or near their rated power levels but not necessarily when lightly loaded. Larger than necessary drives add weight, inertia and friction loss to the system. Underdesigned drives have lower stiffness, can wear rapidly in continuous or in high duty cycle operation or fail due to accidental overloads.

Joint actuation in robots is generally performed by drive mechanisms which interface the actuator (mechanical work source) to the robot links through the joints in an energy-efficient manner. A variety of drive mechanisms are incorporated in practical robots. The transmission ratio of the drive mechanism sets the torque, speed, and inertia relationship of the actuator to the link. Proper placement, sizing, and design of the drive mechanisms set the stiffness, mass, and overall operational performance of the robot. Most modern robots incorporate efficient, overload damage resistant, back-driveable drives.

Direct Drives

The direct drive is kinematically the simplest drive mechanism. In the case of pneumatic or hydraulic actuated robots, the actuator is directly connected between the links. Electric direct-drive robots employ high-torque, low-speed motors directly interfaced to the links. The complete elimination of free play and smooth torque transmission are features of a direct drive. However, there is often a poor dynamic (inertia ratio) match of the actuator to the link requiring a larger, less energy efficient, actuator.

Band Drives

A variant of direct drive is band drive. A thin alloy steel or titanium band is fixed between the actuator shaft and the driven link to produce limited rotary or linear motion. Drive ratios in the order of up to 10 : 1 (10 actuator revolutions for 1 revolution of the joint) can be obtained. Actuator mass is also moved away from the joint – usually toward the base, to reduce robot inertia and gravity loading. It is a smoother and generally stiffer drive than a cable or belt drive.

Belt Drives

Synchronous (toothed) belts are often employed in drive mechanisms of smaller robots and some axes of larger robots. These function much the same as band drives, but have the ability to drive continuously. Multiple stages (two or three) of belts are occasionally used to produce large drive ratios (up to 100 : 1). Tension is controlled with idlers or center adjustment. The elasticity and mass of long belts can cause drive instability and thus increased robot settling time.

Gear Drives

Spur or helical gear drives provide reliable, sealed, low-maintenance power transmission in robots. They are used in robot wrists where multiple axes intersect and compact drive arrangements are required. Large-diameter, turntable, gears are used in the base joints of larger robots to handle high torques with high stiffness. Gears are often used in stages and often with long drive shafts, enabling large physical separation between ac-

tuator and driven joint. For example, the actuator and one stage of reduction may be located near the elbow driving another stage of gearing or differential in a wrist through a long hollow drive shaft (Fig. 3.1).

Planetary gear drives are often integrated into compact gearmotors (Fig. 3.20). Minimizing backlash (free play) in a joint gear drive requires careful design, high-precision and rigid support to produce a drive mechanism which does not sacrifice stiffness, efficiency and accuracy for low backlash. Backlash in robots is controlled by a number of methods including selective assembly, gear center adjustment, and proprietary anti-backlash designs.

Worm Gear Drives

Worm gear drives are occasionally used in low-speed robot manipulator applications. They feature right-angle and offset drive capability, high ratios, simplicity, good stiffness and load capacity. They also have poor efficiency which makes them non-back-driveable at high ratios. This causes the joints to hold their position when unpowered but also makes them prone to damage by attempts to manually reposition the robot.

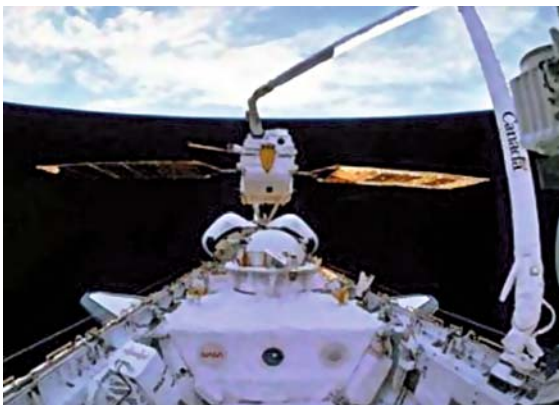


Fig. 3.20 The Space Shuttle robot arm has planetary gear joint drives



Fig. 3.21 The harmonic drive



Fig. 3.22 The Nabtesco RV drive

Proprietary Drives

Proprietary drives are widely used in standard industrial manipulators. The harmonic drive and the rotary vector (RV) drive are two examples of compact, low-backlash, high-torque-capability drives using special gears, cams, and bearings (see Figs. 3.21 and 3.22).

Harmonic drives are frequently used in very small to medium-sized robots. These drives have low backlash, but the flexspline allows elastic windup and low stiffness during small reversing movements. RV drives are usually used in larger robots, especially those subject to overloads and shock loading.

Linear Drives

Direct-drive linear actuators incorporate a linear motor with a linkage to a linear axis. This linkage is often merely a rigid or flexure connection between the actuator forcer and the robot link. Alternatively, a packaged linear motor with its own guideways is mechanically connected directly to a linear axis. Direct linear electromagnetic drives feature zero backlash, high stiffness, high speeds, and excellent performance but are heavy, have poor energy efficiency, and cost more than other types of linear drives.

Ball Screws

Ball-screw-based linear drives efficiently and smoothly convert rotary actuator motion into linear motion. Typically, a recirculating ball nut mates with a ground and hardened alloy steel screw to convert rotary motion into linear motion. Ball screws can be easily integrated into linear axes. Compact actuator/drive packages are available, as well as components for custom integration. Stiffness is good for short and medium travel, however it is lower for long motions because the screw can only be supported at its ends. Low or zero backlash can be obtained with precision-ground screws. Speeds are limited by screw dynamic stability so rotating nuts enable higher speeds. Low-cost robots may employ plain screw drives featuring thermoplastic nuts on smooth rolled thread screws.

3.8 Robot Performance

Industrial robot performance is often specified in terms of functional operations and cycle time. For assembly robots the specification is often the number of typical pick-and-place cycles per minute. Arc-welding robots are specified with a slow weld pattern and weave speed

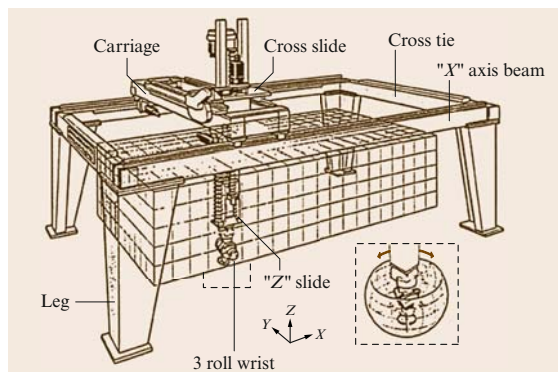


Fig. 3.23 The NASA gantry robot

Rack-and-Pinion Drives

These traditional components are useful for long motions where the guideways are straight or even curved. Stiffness is determined by the gear/rack interface and independent of length of travel. Backlash can be difficult to control as rack-to-pinion center tolerances must be held over the entire length of travel. Dual pinion drives are sometimes employed to deal with backlash by providing active preload. Forces are generally lower than with screws due to lower ratios. Small-diameter (low teeth count) pinions have poor contact ratios, resulting in vibration. Sliding involute tooth contact requires lubrication to minimize wear. These catalog stock drive components are often used on large gantry robots and track-mounted manipulators (Fig. 3.23).

Other Drive Components

Splined shafts, kinematic linkages (four-bar, slider-crank mechanisms, etc.) chains, cables, flex couplings, clutches, brakes, and limit stops are some examples of other mechanical components used in robot drive mechanisms (Fig. 3.8). The Yaskawa RobotWorld assembly and process automation robots are magnetically suspended, translate on air a planar (two-DOF) bearing, and are powered by a direct electromagnetic drive planar motor with no internal moving parts (Fig. 3.12).

as well as by a fast repositioning speed. For painting robots, the deposition or coverage rate and spray pattern speed are important. Peak robot velocity and acceleration catalog data are generally just calculated numbers and will vary due to dynamic (inertia) and static (grav-

ity) coupling between robot joints due to configuration changes as a robot moves.

3.8.1 Robot Speed

Maximum joint velocity (angular or linear) is not an independent value. For longer motions it is often limited by servomotor bus voltage or maximum allowable motor speed. For manipulators with high accelerations, even short point-to-point motions may be velocity limited. For low-acceleration robots, only gross motions will be velocity limited. Typical peak end-effector speeds can range up to 20 m/s for large robots.

3.8.2 Robot Acceleration

In most modern manipulators, because the payload mass is small when compared with the manipulator mass, more power is spent accelerating the manipulator than the load. Acceleration affects gross motion time as well as cycle time (gross motion time plus settling time). Manipulators capable of greater acceleration tend to be stiffer manipulators. In high-performance robot manipulators, acceleration and settling time are more important design parameters than velocity or load capacity. Maximum acceleration for some assembly and material handling robots is in excess of 10 g with light payloads.

3.8.3 Repeatability

This specification represents the ability of the manipulator to return repeatedly to the same location. Depending on the method of teaching or programming the manipulator, most manufacturers intend this figure to indicate the radius of a sphere enclosing the set of locations to which the arm returns when sent from the same origin by the same program with the same load and setup conditions. This sphere may not include the target point because calculation round-off errors, simplified calibration, precision limitations, and differences during the teaching and execution modes can cause significantly larger errors than those just due to friction, unresolved joint and drive backlash, servo system gain, and structural and mechanical assembly clearances and play. The designer must seriously consider the real meaning of the required repeatability specification. Repeatability is important when performing repetitive tasks such as blind assembly or machine loading. Typical repeatability specifications range from 1–2 mm for large spot-welding robots to 0.005 mm (5 μm) for precise micropositioning manipulators.

3.8.4 Resolution

This specification represents the smallest incremental motion that can be produced by the manipulator. Resolution is important in sensor-controlled robot motion and in fine positioning. Although most manufacturers calculate system resolution from the resolution of the joint position encoders, or from servomotor and drive step size, this calculation is misleading because system friction, windup, backlash, and kinematic configuration affect the system resolution. Typical encoder or resolver resolution is 10^{14} – 10^{25} counts for full-axis or joint travel, but actual physical resolution may be in the range 0.001–0.5 mm. The useful resolution of a multi-joint serial-link manipulator is worse than that of its individual joints.

3.8.5 Accuracy

This specification covers the ability of a robot to position its end-effector at a preprogrammed location in space. Robot accuracy is important in the performance of non-repetitive types of tasks programmed from a database, or for taught tasks that have been remapped or offset owing to measured changes in the installation.

Accuracy is a function of the precision of the arm kinematic model (joint type, link lengths, angles between joints, any accounting for link or joint deflections under load, etc.), the precision of the world, tool, and fixture models, and the completeness and accuracy of the arm solution routine. Although most higher-level robot programming languages support arm solutions, these solutions usually model only simplified rigid-body kinematic configurations. Thus, manipulator accuracy becomes a matter of matching the robot geometry to the robot solution in use by precisely measuring and calibrating link lengths, joint angles, and mounting positions.

Typical accuracies for industrial manipulators range from ± 10 mm for uncalibrated manipulators that have poor computer models to ± 0.01 mm for machine-tool-like manipulators that have controllers with accurate kinematic models and solutions and precisely manufactured and measured kinematic elements.

3.8.6 Component Life and Duty Cycle

The three subassemblies in an electrically powered robot with the greatest failure problems are the actuators (servomotors), transmissions, and power and signal cables. Mean time between failures (MTBF) should be

a minimum of 2000 h on line, and ideally at least 5000 operating hours should pass between major component preventive maintenance replacement schedules.

Worst-case motion cycles must be assumed as most current robot installations are used in generally repetitive tasks. Small-motion design-cycle life (less than 5% of joint travel range) for assembly robots should be 20–100 million full bidirectional cycles. Large-motion cycle life (greater than 50% of full joint range) should typically be 5–40 million cycles.

Short-term peak performance is frequently limited by maximum drive loading, whereas long-term, continuous, performance is limited by motor heating. Rather than design for equal levels of short- and long-term performance, cost savings and performance improvements can result from designing for an anticipated duty cycle. This allows the use of smaller, lower-inertia, lighter motors. Industrial robots usually become obsolete and are replaced before they reach their design cycle life.

3.8.7 Collisions

In the course of operation, unforeseen or unexpected situations may occasionally result in a collision involving the manipulator, its tools, the workpiece, or other objects in the workplace. These accidents may result in no, little, or extensive damage, depending in large part on the design of the manipulator. Crash-resistant design options should be considered early in the design process if the time lost or cost of such accidents could be significant. Typical damage due to accidents include fracture or shear failures of gear teeth or shafts, dented or bent link structures, slipping of gears or pulleys on shafts, cut or severely abraded or deformed wires, cables or hoses, and broken connectors, fittings, limit stops or switches. Compliant elements such as overload (slip) clutches, elastic members, and padded surfaces can be incorporated to reduce shock loads and help decouple or isolate the actuators and drive components in the event of such collisions.

3.9 Conclusions and Further Reading

The mechanical design of a robot is an iterative process involving engineering, technical, and application-specific considerations evaluations, and choices. The final design should reflect consideration of detailed task requirements rather than simply broad specifications. Proper identification and understanding of these requirements is a key to achieving the design goals.

Design and choice of specific components involves tradeoffs. A purely static, rigid-body approach to manipulator design is often used, but is not always sufficient. Mechanical system stiffness, natural frequencies, control system compatibility, and intended robot applications and installation requirements must be considered.

There are many opportunities for further reading on the design of the mechanisms and actuation that form the core of a robotic system. A well-known and useful reference for robot design is *Rivin* [3.44].

Craig [3.6] and *Tsai* [3.9] provide the mathematical relations between the mechanical structure of a robot and its workspace and mechanical advantage. *Sclater* and *Chironis* [3.48] is a reprint of a valuable compendium of devices useful for a variety of applications, such as joint drives and transmissions. See *McCarthy* [3.34] for geometric techniques to design specialized mechanisms.

Juvinall and *Marshek* [3.45] and *Shigley* and *Mishke* [3.46,47] are important references for the design of the components such as the link structure, bearings, and transmissions that are central to the effective mechanical performance of robotic systems.

Although many design decisions can be made through the application of straightforward algorithms and equations, a multitude of other important considerations transform the challenge of robot design into one requiring good engineering judgment.

References

- | | |
|---|---|
| <p>3.1 D.T. Greenwood: <i>Classical Dynamics</i> (Prentice-Hall, Upper Saddle River 1977)</p> <p>3.2 F.C. Moon: <i>Applied Dynamics</i> (Wiley-Interscience, New York 1998)</p> | <p>3.3 S.-M. Song, K.J. Waldron: <i>Machines that Walk: The Adaptive Suspension Vehicle</i> (MIT Press, Cambridge 1988)</p> |
|---|---|

- 3.4 M.T. Mason, J.K. Salisbury: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
- 3.5 R.P. Paul: *Robot Manipulators: Mathematics, Programming, and Control* (MIT Press, Cambridge 1981)
- 3.6 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Addison-Wesley, Publ., Reading 1989)
- 3.7 O. Bottema, B. Roth: *Theoretical Kinematics* (North-Holland, New York 1979), (reprinted by Dover, New York)
- 3.8 J.M. McCarthy: *An Introduction to Theoretical Kinematics* (MIT Press, Cambridge 1990)
- 3.9 L.W. Tsai: *Robot Analysis, The Mechanics of Serial and Parallel Manipulators* (Wiley, New York 1999)
- 3.10 T. Lozano-Perez: Spatial Planning: A configuration space approach, *IEEE Trans. Comput.* **32**(2), 108–120 (1983)
- 3.11 J.C. Latombe: *Robot Motion Planning* (Kluwer Academic, Boston 1991)
- 3.12 R. Vijaykumar, K. Waldron, M.J. Tsai: Geometric optimization of manipulator structures for working volume and dexterity. In: *Kinematics of Robot Manipulators*, ed. by J.M. McCarthy (MIT Press, Cambridge 1987) pp. 99–111
- 3.13 K. Gupta: On the nature of robot workspace. In: *Kinematics of Robot Manipulators*, ed. by J.M. McCarthy (MIT Press, Cambridge 1987) pp. 120–129
- 3.14 I. Chen, J. Burdick: Determining task optimal modular robot assembly configurations, *Proc. IEEE Robot. Autom. Conf.* (1995) pp. 132–137
- 3.15 P. Chedmail, E. Ramstei: Robot mechanisms synthesis and genetic algorithms, *Proc. IEEE Robot. Autom. Conf.* (1996) pp. 3466–3471
- 3.16 P. Chedmail: Optimization of multi-DOF mechanisms. In: *Computational Methods in Mechanical Systems*, ed. by J. Angeles, E. Zakhariiev (Springer, Berlin 1998), pp. 97–129
- 3.17 C. Leger, J. Bares: Automated Synthesis and Optimization of Robot Configurations, CD-ROM Proc. ASME DETC'98 (Atlanta 1998), paper no. DETC98/Mech-5945
- 3.18 F.C. Park: Distance metrics on the rigid body motions with applications to mechanism design, *ASME J. Mech. Des.* **117**(1), 48–54 (1995)
- 3.19 J.M.R. Martinez, J. Duffy: On the metrics of rigid body displacements for infinite and finite bodies, *ASME J. Mech. Des.* **117**(1), 41–47 (1995)
- 3.20 M. Zefran, V. Kumar, C. Croke: Choice of Riemannian metrics for rigid body kinematics, CD-ROM Proc. ASME DETC'96 (Irvine 1996), paper no. DETC96/Mech-1148
- 3.21 Q. Lin, J.W. Burdick: On well-defined kinematic metric functions, *Proc. Int. Conf. on Robotics and Automation* (San Francisco 2000) pp. 170–177
- 3.22 C. Gosseli: On the design of efficient parallel mechanisms. In: *Computational Methods in Mechanical Systems*, ed. by J. Angeles, E. Zakhariiev (Springer, Berlin, Heidelberg 1998), pp. 68–96
- 3.23 J.V. Albro, G.A. Sohl, J.E. Bobrow, F. Park: On the computation of optimal high-dives, *Proc. Int. Conf. on Robotics and Automation* (San Francisco 2000) pp. 3959–3964
- 3.24 G.E. Shilov: *An Introduction to the Theory of Linear Spaces* (Dover, New York 1974)
- 3.25 J.K. Salisbury, J.J. Craig: Articulated hands: Force control and kinematic issues, *Int. J. Robot. Res.* **1**(1), 4–17 (1982)
- 3.26 J. Angeles, C.S. Lopez-Cajun: Kinematic isotropy and the conditioning index of serial manipulators, *Int. J. Robot. Res.* **11**(6), 560–571 (1992)
- 3.27 J. Angeles, D. Chabla: On isotropic sets of points in the plane. Application to the design of robot architectures. In: *Advances in Robot Kinematics*, ed. by J. Lenarčič, M.M. Stanišić (Kluwer Academic, Dordrecht 2000) pp. 73–82
- 3.28 E.F. Fichter: A Stewart platform-based manipulator: General theory and practical construction. In: *Kinematics of Robot Manipulators*, ed. by J.M. McCarthy (MIT Press, Cambridge 1987) pp. 165–190
- 3.29 J.P. Merlet: *Parallel Robots* (Kluwer Academic, Dordrecht 1999)
- 3.30 J.M. Hervé: Analyse structurelle des mécanismes par groupe des déplacements, *Mechanism Machine Theory* **13**(4), 437–450 (1978)
- 3.31 J.M. Hervé: The Lie group of rigid body displacements, a fundamental tool for mechanism design, *Mechanism Machine Theory* **34**, 719–730 (1999)
- 3.32 A.P. Murray, F. Pierrot, P. Dauchez, J.M. McCarthy: A planar quaternion approach to the kinematic synthesis of a parallel manipulator, *Robotica* **15**(4), 361–365 (1997)
- 3.33 A. Murray, M. Hanchak: Kinematic synthesis of planar platforms with RPR, PRR, and RRR chains. In: *Advances in Robot Kinematics*, ed. by J. Lenarčič, M.M. Stanišić (Kluwer Academic, Dordrecht 2000) pp. 119–126
- 3.34 J.M. McCarthy: *Geometric Design of Linkages* (Springer, Berlin, Heidelberg 2000)
- 3.35 V. Kumar: Instantaneous kinematics of parallel-chain robotic mechanisms, *J. Mech. Des.* **114**(3), 349–358 (1992)
- 3.36 C. Gosselin, J. Angeles: The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator, *ASME J. Mech. Transmiss. Autom. Des.* **110**(3), 35–41 (1988)
- 3.37 J. Lee, J. Duffy, M. Keler: The optimum quality index for the stability of in-parallel planar platform devices, CD-ROM Proc. 1996 ASME Design Engineering Technical Conferences (Irvine 1996), 96-DETC/MECH-1135
- 3.38 J. Lee, J. Duffy, K. Hunt: A practical quality index based on the octahedral manipulator, *Int. J. Robot. Res.* **17**(10), 1081–1090 (1998)
- 3.39 S.E. Salcudean, L. Stocco: Isotropy and actuator optimization in haptic interface design, *Proc. Int. Conf.*

- on Robotics and Automation (San Francisco 2000) pp. 763–769
- 3.40 L.-W. Tsai, S. Joshi: Kinematics and optimization of a spatial 3-UPU parallel manipulator, *J. Mech. Des.* **122**, 439–446 (2000)
- 3.41 Q. Jin, T.-L. Yang: Theory for topology synthesis of parallel manipulators and its application to three-dimension-translation parallel manipulators, *J. Mech. Des.* **126**(3), 625–639 (2004)
- 3.42 X. Kong, C.M. Gosselin: Type synthesis of three-degree-of-freedom spherical parallel manipulators, *Int. J. Robot. Res.* **23**, 237–245 (2004)
- 3.43 T.A. Hess-Coelho: Topological synthesis of a parallel wrist mechanism, *J. Mech. Des.* **128**(1), 230–235 (2006)
- 3.44 E.I. Rivin: *Mechanical Design of Robots* (McGraw-Hill, New York 1988) p. 368
- 3.45 R.C. Juvinall, K.M. Marshek: *Fundamentals of Machine Component Design*, 4th edn. (Wiley, New York 2005) p. 832
- 3.46 J.E. Shigley, C.R. Mischke: *Mechanical Engineering Design*, 7th edn. (McGraw-Hill, Upper Saddle River 2004) p. 1056
- 3.47 J.E. Shigley, C.R. Mischke: *Standard Handbook of Machine Design*, 2nd edn. (McGraw-Hill, Upper Saddle River 1996) p. 1700
- 3.48 N. Sclater, N. Chironis: *Mechanisms and Mechanical Devices Sourcebook*, 4th edn. (McGraw Hill, Upper Saddle River 2007) p. 512
- 3.49 J.P. Trevelyan: Sensing and control for shearing robots, *IEEE Trans. Robot. Autom.* **5**(6), 716–727 (1989)
- 3.50 J.P. Trevelyan, P.D. Kovesi, M. Ong, D. Elford: ET: A wrist mechanism without singular positions, *Int. J. Robot. Res.* **4**(4), 71–85 (1986)

Sensing and

4. Sensing and Estimation

Henrik I. Christensen, Gregory D. Hager

Sensing and estimation are essential aspects of the design of any robotic system. At a very basic level, the state of the robot itself must be estimated for feedback control. At a higher level, perception, which is defined here to be task-oriented interpretation of sensor data, allows the integration of sensor information across space and time to facilitate planning.

This chapter provides a brief overview of common sensing methods and estimation techniques that have found broad applicability in robotics. The presentation is structured according to a process model that includes sensing, feature extraction, data association, parameter estimation, and model integration. Several common sensing modalities are introduced and characterized. Common methods for estimation in linear and nonlinear systems are discussed, including statistical estimation, the Kalman filter, and sample-based methods. Strategies for robust estimation are also briefly described. Finally,

4.1	The Perception Process	88
4.2	Sensors	90
4.3	Estimation Processes	93
4.3.1	Point Estimation	94
4.3.2	Other Approaches to Estimation	97
4.3.3	Robust Estimation Methods	98
4.3.4	Data Association Techniques	100
4.3.5	Modeling Sensors	102
4.3.6	Other Uncertainty Management Methods	103
4.4	Representations	104
4.4.1	Raw Sensor Representations	104
4.4.2	Grid-Based Representations	104
4.4.3	Discrete Feature Representations ...	105
4.4.4	Symbolic/Graph-Based Models	105
4.5	Conclusions and Further Readings	106
	References	106

several common representations for estimation are introduced.

Controlling a robotic system would be relatively simple if a complete model of the environment was available, and if the robot actuators could execute motion commands perfectly relative to this model. Unfortunately, in most cases of interest a complete world model is not available, and perfect control of mechanical structures is never a realistic assumption. Sensing and estimation are a means of compensating for this lack of complete information. Their role is to provide information about the state of the environment and the state of the robot system as a basis for control, decision making, and interaction with other agents in the environment, such as humans.

For the purposes of discussion, we will differentiate between sensing and estimation to recover the state of the robot itself, referred to as *proprioception*, versus sensing and estimation to recover the state of the external

world, referred to as *exteroception*. In practice, most robot systems are designed to have the proprioception necessary to estimate and control their own physical state. On the other hand, recovering the state of the world from sensor data is usually a much larger and more complex problem.

Early work on computational perception for robotics assumed that one could recover a complete general-purpose model of the environment, use such a model to make decisions, and subsequently act on them, as for example presented by [4.1]. More recently it has become apparent that such an approach is not realistic. Indeed, considering that sensor-based robots now appear in diverse applications such as mobile surveillance, high-performance manipulation, and medical interventions, it is clear that appropriate sensing and estimation for a given system must be highly task dependent. Conse-

quently, the discussion here is organized along the lines of task-oriented sensing and estimation of the external world.

Sensing and estimation together can be viewed as the process of transforming a physical quantity into a computer representation that can be used for further processing. Sensing is thus closely tied to transducers that transform some physical entity into a signal that can be processed by a computer. Sensing is also intimately tied to perception, the process of representing the sensory information in an task-oriented model of the world. However, sensor data is usually corrupted in various ways that complicate this process. Statistical noise arises from the transducer, discretization is introduced in the digitization process, and ambiguity is introduced by poor sensor selectivity to name a few examples.

Estimation methods are thus introduced to support appropriate integration of information into models of the environment and for improvement of the signal-to-noise ratio.

In this chapter the general characteristics of sensing and estimation are introduced, while more in-depth presentations of select topics are provided in Part C of the handbook. In Sect. 4.1 the overall sensing/perception process is introduced. In Sect. 4.2 different kinds of sensors are introduced and some key characteristics are presented. Estimation of world representations can utilize a number of different methods involving both parametric and nonparametric techniques as discussed in Sect. 4.3. For model-based integration a variety of different representations can be used, as described in Sect. 4.4.

4.1 The Perception Process

The input to the perception process is typically twofold: (1) digital data from a number of sensors/transducers, and (2) a partial model of the environment (a world model) that includes information about the state of the robot and other relevant entities in the external world. The sensor data itself can take on a number of different forms such as a scalar or vector value $x(\alpha, \beta)$ acquired over a time series $x(t)$, a scan $x_t(\theta_i)$, a vector field x or a three-dimensional volume $x(\rho, \theta, \phi)$. In many cases, a system must integrate data from several disparate sensors, for example, an estimate of the position of a mobile robot may integrate data from axis encoders, vision, global positioning system (GPS) data, and inertial sensors.

To further structure the discussion in this chapter, we adopt a general model of the perception process as shown in Fig. 4.1. In this model, we have included the most common operations applied to integrate sensor data with

a world model. Depending on the task in question, some of the included modules may be missing, and others may themselves take on a complicated structure. However, the supplied model suffices to illustrate many of the issues in sensing and estimation. In the remainder of this section, we discuss an example from mobile localization to illustrate this model.

The initial problem in sensory processing is data preprocessing and feature extraction. The role of preprocessing is to reduce noise from the transducer, to remove any systematic errors, and to enhance relevant aspects of the data. In some cases, sensory information might also have to be temporally or spatially aligned for subsequent integration. There are innumerable ways that data can be preprocessed to enhance or extract features that are used in the integration. One common approach is model fitting, as illustrated for a laser scanner in Fig. 4.2. Once sensor information is available,

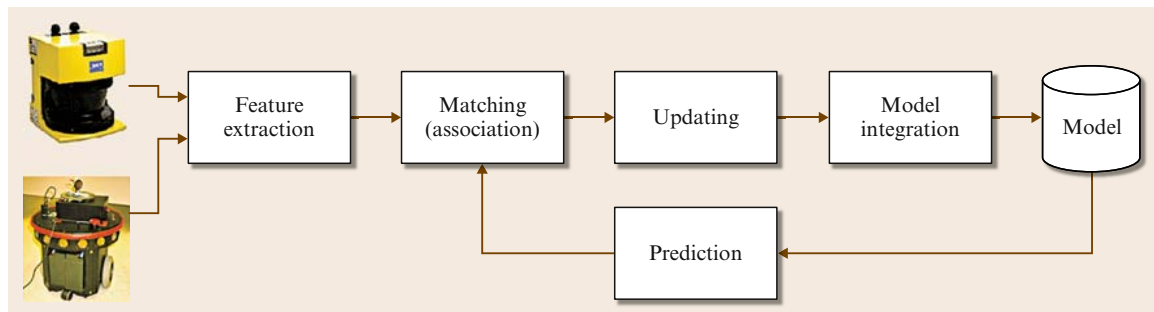


Fig. 4.1 Example of a perception process as discussed in this chapter

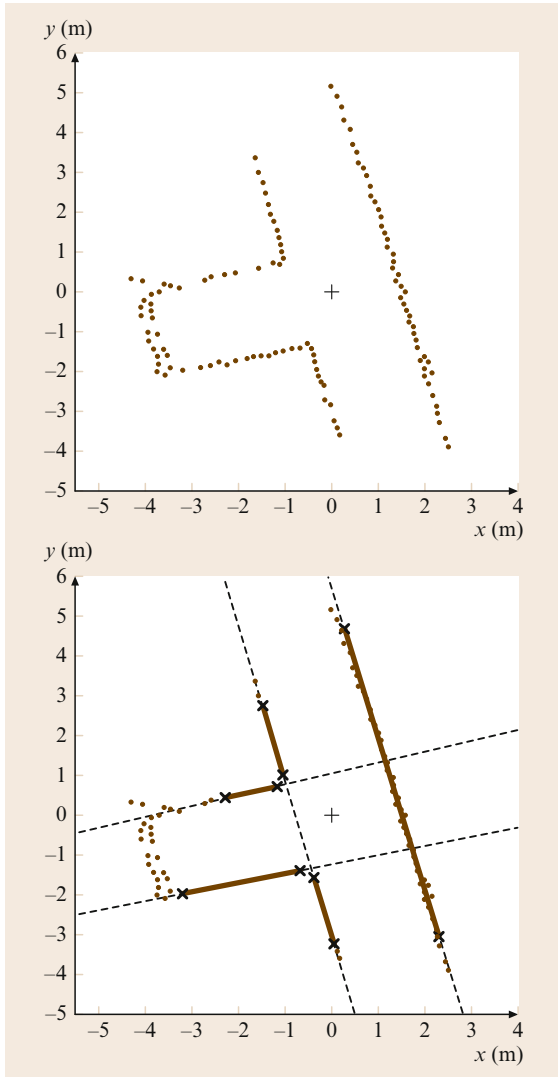


Fig. 4.2 An example of feature extraction from a laser scan (after [4.2])

it is often necessary to match the data with an existing model (Fig. 4.3). This model may be based on a priori known structure [e.g., a computer-aided design (CAD) model of the environment], or may have been built up from previously acquired data. Data association methods are commonly employed to estimate the relationship between sensor data and the model of the en-

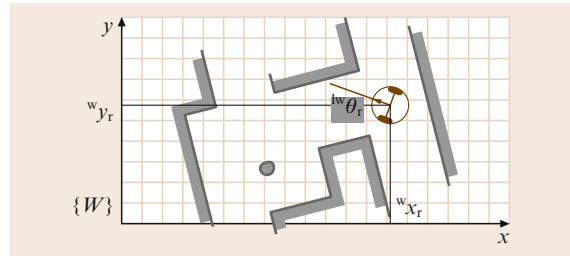


Fig. 4.3 An example environmental model for mobile robot localization (after [4.2])

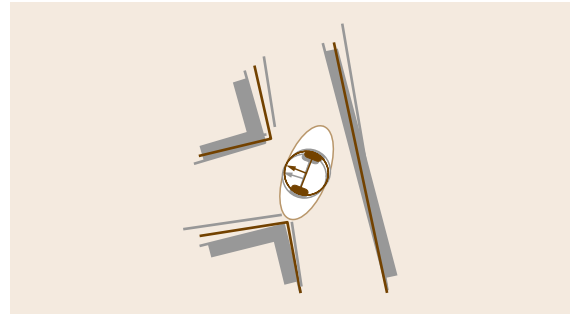


Fig. 4.4 Estimation of position and orientation for the example mobile robot (after [4.2])

vironment. In our mobile robot localization example, the extracted line features are matched against a polygonal world model. This matching process can be performed in several different ways, but in general it is an optimization that maximizes the alignment of features to the model.

Once sensory data has been matched against the world model it is possible to update the model with new information contained in the sensor data. In the example, the orientation and position of the robot relative to the world model can be updated (Fig. 4.4) from the matched line segments.

Finally, it may be possible to develop a dynamical system model of the underlying state being estimated. Using such a system model, it is possible to predict how the world changes over time until new sensory data is acquired. This can be used within a feed-forward prediction process, which in turn simplifies data association for new sensory readings, as shown in Fig. 4.1.

With this as a prologue, we now turn to discuss each step of the perception process in greater detail.

4.2 Sensors

There are a variety of ways to classify sensors depending on what they measure, and how they measure it. As noted previously, proprioceptive sensors are used to measure the internal state of a robot, which might include position of different degrees of freedom, temperature, voltage on key components, motor current, force applied to an effector, and so forth. Exteroceptive sensors, on the other hand, generate information about the external environment in terms of distance to an object, interaction forces, tissue density, and so forth.

Sensors may also be differentiated based on whether they are passive or active. In general, an active sensor is one that emits energy into the environment, and measures properties of the environment based on the response. A passive sensor is one that is not active. Active sensors are generally more robust than passive sensors since they exert some control over the measured signal. For example, a passive stereo camera system must rely on the appearance of viewed surfaces when performing feature matching for triangulation (Chap. 22), whereas structured light systems project a pattern onto the scene and are thus less sensitive to scene characteristics. Even so, absorption, scattering or interference of the emitted signal can affect the performance of active sensors.

Proprioceptive sensors are typically passive and usually measure physical properties of the robot such as joint position, velocity, or acceleration, motor torque, and so forth. Exteroceptive sensors, on the other hand, can be further divided into contact and noncontact sensing. The contact sensors are typically the same modalities as used for proprioception, while noncontact sensor sensors involve most of the modalities that can be used for estimation of physical properties at a distance including intensity, range, direction, size, and so forth.

A classification of typical sensors according to method and typical application is shown in Table 4.1. More detail on methods of sensing, characterization of

sensors, and general applications can for example be found in the Handbook of Modern Sensors [4.3] and in Part C of this handbook.

Table 4.1 Classification of sensors frequently used in robotics according to sensing objective [proprioception (PC)/exteroception (EC)] and method (active/passive)

Classification	Sensor type	Sens	A/P
Tactile sensors	Switches/bumpers	EC	P
	Optical barriers	EC	A
	Proximity	EC	P/A
Haptic sensors	Contact arrays	EC	P
	Force/torque	PC/EC	P
	Resistive	EC	P
Motor/axis sensors	Brush encoders	PC	P
	Potentiometers	PC	P
	Resolvers	PC	A
	Optical encoders	PC	A
	Magnetic encoders	PC	A
	Inductive encoders	PC	A
	Capacity encoders	EC	A
Heading sensors	Compass	EC	P
	Gyroscopes	PC	P
	Inclinometers	EC	A/P
Beacon based (postion wrt an inertial frame)	GPS	EC	A
	Active optical	EC	A
	RF beacons	EC	A
	Ultrasound beacon	EC	A
	Reflective beacons	EC	A
Ranging	Capacitive sensor	EC	P
	Magnetic sensors	EC	P/A
	Camera	EC	P/A
	Sonar	EC	A
	Laser range	EC	A
	Structures light	EC	A
Speed/motion	Doppler radar	EC	A
	Doppler sound	EC	A
	Camera	EC	P
	Accelerometer	EC	P
Identification	Camera	EC	P
	Radio frequency identification		
	RFID	EC	A
	Laser ranging	EC	A
	Radar	EC	A
	Ultrasound	EC	A
	Sound	EC	P

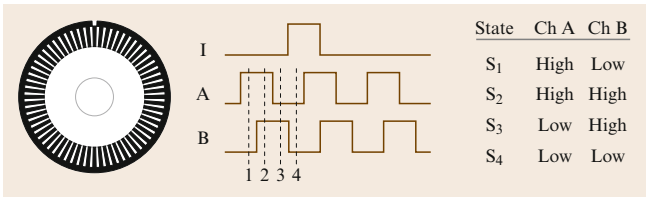


Fig. 4.5 Sketch of the quadrature encoder disc, and output from photodetectors placed over each of the two pattern. The corresponding state changes are shown on the right

Estimation of rotational motion is fundamental to control of robot manipulators and also for estimation of ego-motion for mobile systems. The most common sensor for measurement of rotation is the quadrature encoder. It is composed of a transparent disc, with two periodic patterns that are out of phase, as shown in Fig. 4.5. Through the use of counters it is possible to directly compute the motion and its direction (the phasing between sensors A and B in Fig. 4.5). In addition the disc is frequently fitted with a single dot on the outer rim for indexing (specification of a zero index). The density of the pattern determines the resolution of the measurements. When fitting the sensor to a motor before a reduction gear it is easy to achieve accuracies beyond $1/1000^\circ$.

For the estimation of force and torque at an end-effector it is possible to use piezoelectric elements. These elements generate a voltage that is proportional to the introduced deformation. Through careful placement it is possible to measure both force and torque. The sensors are used in robotic manipulation as part of assembly systems, deburring, etc. and also in medical applications for the estimation of stress and contact. Force/torque sensors are widely available in a range of sizes and dy-

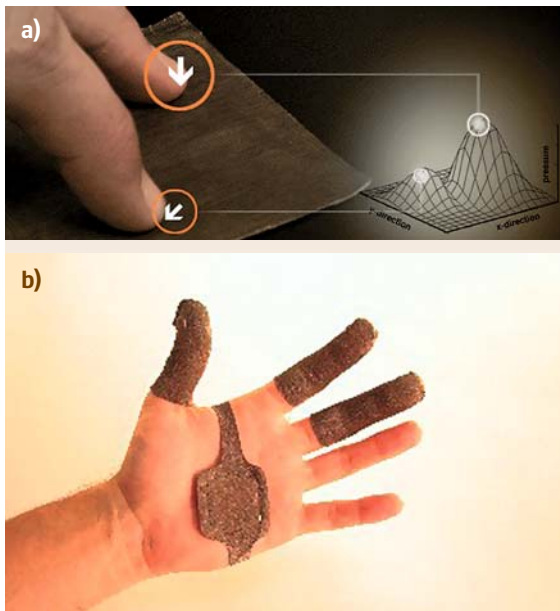


Fig. 4.6 (a) TactArray, a flexible capacitive array tactile sensor from Pressure Profile Systems, Inc., is appropriate for sensing contact locations and areas under sliding conditions. (b) Conformable TactArray sensors can fit on a human or robotic hand (courtesy Pressure Profile Systems, Inc.)

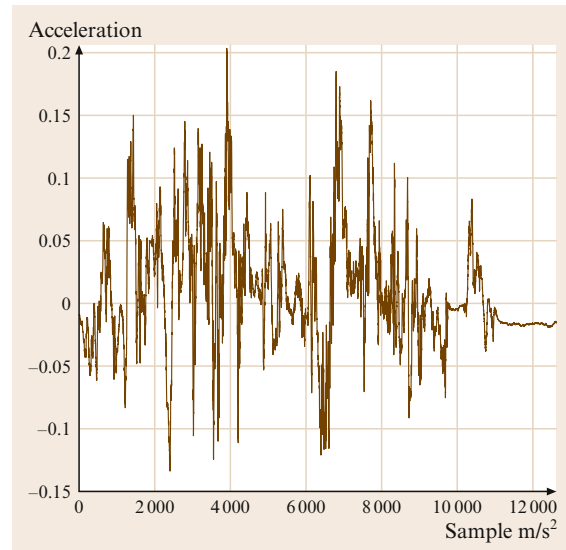


Fig. 4.7 Example data from an IMU unit for driving on an unpaved road

namic ranges, including new flexible arrays that can be mounted on a variety of end-effectors (Fig. 4.6). Potential problems with force sensors are a dead band on initial contact, and noisy data from the basic sensing elements, which calls for signal processing to clean up the data.

Ego-motion estimation is an important part of almost all robotic systems. To this end it is possible to use inertial measurement units (IMU). An IMU typically includes both accelerometers and gyros. Accelerometers are sensitive to all types of acceleration, which implies that both translation motion and rotation (centripetal forces) are measured in combination. Joint IMU units allow the estimation of rotation and translation, and allow for double integration to estimation the velocity, orientation, and position of a system, as for example reported in [4.4]. One of the problems associated with the use of an IMU is the need for double integration. Small biases and noise can result in significant divergence in the final estimate, which calls for use of detailed models and careful calibration and identification of sensor characteristics. An example of data from a cross-bow DMU-6x unit for a car driving on an unpaved road is shown in Fig. 4.7.

Much early work on mobile robotics, underwater robots, and some medical robotics relies on ultrasonic ranging. The general class of sensors are often termed sound navigation and ranging (sonars). The general principle is that the systems emits a sound pulse and awaits the return of echoes that have bounced off objects in



Fig. 4.8 Example laser ranging sensor (SICK LMS291) that is widely used in mobile robotics

the environment. Knowing the transmission speed in the medium and the time of flight it is possible to compute the distance. The method was widely used in early robotics due to the availability of low-cost sensors with adequate performance. In underwater robotics this is still a primary sensor. The sonar is discussed in detail in Chap. 21.

Recent progress, in particular on environmental modeling and navigation, has in many respects been due to the emergence of low-cost high-fidelity laser scanning systems. The SICK series of laser scanners are time-of-flight scanners. The scanner sends out a pulse of light and measures the time to return. The standard scanner enables estimation of distances up to 80 m at centimeter or millimeter accuracy. The scanner measures distances in a plane with an angular resolution of 0.5° . The field of view is 180° resulting in 181–361 range measurements. The sensor data are contaminated by uniformly distributed noise, which must be considered in the detection of features or integration of data into a raw sensor map.

Imaging sensors are a rich source of information for sensing and estimation. Imaging sensors come in a wide variety of configurations, varying according to imaging geometry, image resolution, sensor technology and the range of sensed spectral bands. Most readers are no doubt familiar with the traditional three-**CCD**, perspective color camera. In this case, there are three charge-coupled detector (**CCD**) arrays, each receiving a portion of the visible spectrum corresponding roughly to the human perception of red, green, and blue colors. A common and less expensive alternative is a so-called single-chip **CCD** camera. In this case, a special spatial array of color filters, usually referred to as a Bayer filter after its inventor Bryce Bayer, is employed. The result-

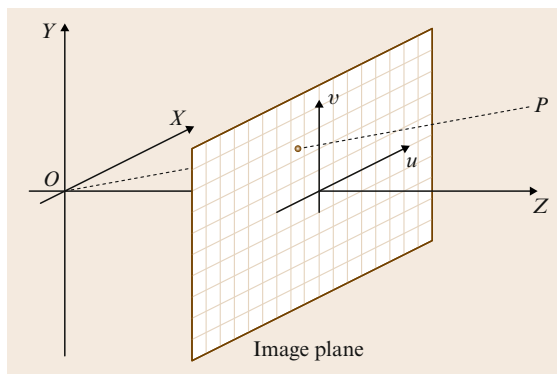


Fig. 4.9 The pinhole camera model

ing spatial array is subsequently processed (a process referred to as demosaicing) to provide color information for each pixel.

In the United States, image sensors traditionally contained 480 rows of 640 pixels according to the NTSC standard created for analog transmission of television signals. The corresponding European standard, PAL, has 576 lines of 768 pixels. More recently, the advent of



Fig. 4.10 Catadioptric image and the same image mapped to a cylindrical surface

digital interfaces such as IEEE 1394 and USB 2.0 have allowed camera systems to be developed with significantly improved resolution ranging into the millions of pixels. At the same time, cost-effective infrared (IR) and ultraviolet (UV) cameras have become available, allowing the development of advanced multispectral image interpretation systems.

A traditional imaging sensor contains an optical system that focuses light on a planar imaging array. In most cases, this system can be modeled using the classical pinhole camera model (shown in Fig. 4.9). Given a point $(x, y, z)^T$ in Euclidean space, the corresponding camera pixel coordinates $(u, v)^T$ are given by

$$\begin{aligned}(u - u_c) &= \frac{f}{s_x} \frac{x}{z}, \\ (v - v_c) &= \frac{f}{s_y} \frac{y}{z},\end{aligned}\quad (4.1)$$

where f is the focal length of the lens system, u_c and v_c are the pixel coordinates of the center of projection, and s_x and s_y are the size of a single pixel on the imaging array. In practice, these models are also augmented with low-order models of image distortion.

4.3 Estimation Processes

As discussed in the introduction, there are many different techniques for combining information from sensors. The appropriate set of techniques depends, to a great degree, on what is known a priori about the environment, what information is necessary for the task at hand, and what models for the sensing system are appropriate. Common methodologies include simple voting-based methods, parametric and nonparametric statistical estimation techniques, fuzzy logic-based systems, and Dempster–Shafer theory.

To illustrate this point, consider the robot localization problem introduced in Sect. 4.1. At the outset, if nothing is known about the environment, the robot may acquire a laser scan and try to produce an initial model of the environment using line segments. Since nothing is known a priori, the system must estimate: (1) the number of line segments, (2) the data association between line segments and observed data values, and (3) the parameters of the line segments themselves. This is a challenging problem that can be attacked by simple voting techniques such as the Hough transform [4.8] or RANSAC [4.9] or more sophisticated unsupervised clustering methods such as k -means [4.10], expecta-

The values of these parameters for a given camera system can be determined experimentally using a variety of methods [4.5].

By combining a traditional perspective camera with a mirror, creating a so-called catadioptric system, it is possible to create imaging geometries that map fields of view as large as a hemisphere into a single image. Such systems are useful, for example, for surveillance, and their geometric properties provide for stable position referencing for mobile navigation [4.6]. An example image is shown in Fig. 4.10 together with the corresponding image when mapped to a cylindrical surface.

The discussion above has touched on the most commonly employed robotic sensing devices. Many special-purpose sensors are employed for specific applications. In medicine (Chap. 52), ultrasound, X-ray, computed tomography, and magnetic resonance imaging are commonly employed. Underground mapping makes use of ground-penetrating radar [4.7]. Underwater robotics makes use of many variations on acoustical sensors. Further discussions of these more task-specific sensing modalities can be found in the application chapters in Part C of this handbook.

tion maximization (EM) [4.11], or GPCA [4.12]. In many cases, this is a computationally intensive, iterative process.

Conversely, if a prior CAD model for the environment is known, then the problem is to produce a small set of parameters (translation and rotation) of the model to match the data. This problem can be solved, using feature matching by aligning observed points to the model with iterative closest-point algorithms (ICP) [4.13] or other efficient combinatorial matching algorithms such as Monte Carlo methods [4.14]. The best method to apply again depends to a great degree on the structure of the environment and what is known a priori.

Once an initial registration is known, new data can take advantage of the fact that strong prior knowledge is available. In particular, as the robot moves, the sensor data should change in a predictable fashion. Thus it is possible to make use of predictor–corrector methods such as the Kalman filter [4.15, 16] or sequential importance sampling [4.17], provided appropriate statistical characterizations of the sensing system are available. The data association problem, if present, can be addressed using a variety of general techniques such as

EM [4.10] or more specialized modifications to the previously cited predictor–corrector methods [4.18].

It is often the case that sensor data is corrupted by occasional nonsensical values. For example, the laser range finder in our example may occasionally return a spurious range value due to a reflection. Many commonly used estimation techniques are not robust to such so-called data outliers. Techniques from robust statistics [4.19] can be used to improve the performance of sensing and estimation systems in such cases.

Finally, we may want to consider what information is actually important for the task at hand. Most of the techniques above presume that the goal is to produce an accurate estimate of a set of continuous parameters closely related to the underlying data. However, in some tasks, the parameter values themselves may not be what is of interest. For example, suppose that the goal of our robot is to drive through a doorway. Although this clearly depends on an ability to estimate the width of the door (a continuous parameter), the decision is ultimately binary. This problem can be codified as a *decision problem*. Decision problems can be modeled using concepts from *decision theory* [4.20] including zero–one loss functions, likelihood ratios, or probability ratios. For example, in the case of fitting through a door, for a low-priority task there may be a low cost associated with not attempting to move through this particular door (necessitating replanning to find an alternative route) relative to attempting to navigate through an opening that is too small (risking damage to the robot or the door, or both). Conversely, if the task is urgent, more risky behavior may be warranted.

For any given task (or decision), the amount of information necessary to reach the decision may vary, for example, if the doorway is quite wide, it may require relatively little information to safely navigate through it. Conversely, a tight fit may require close inspection before a decision can be reached. The problem of determining the type and/or amount of information necessary to reach a decision is referred to variously as the *sequential sampling problem* [4.20], the sensor control problem, or the sensor planning problem [4.21–23].

4.3.1 Point Estimation

In our robot localization example we saw several cases where the key problem was to estimate an unknown quantity that can be represented as a point in a vector space. Examples include the location of a two-dimensional (2-D) or three-dimensional (3-D) point or the location of a robot. We also saw examples where

the problem was to locate the pose (position and orientation) of the robot, or parameters of a line segment. The latter differ in that the underlying parameter space is not a vector space. This introduces some additional unique problems. We refer the reader to [4.24, 25] for further discussion, and in the remainder of this chapter restrict our attention to *point estimation* problems on vector spaces. In our discussion, we assume the reader is familiar with multivariate Gaussian distributions as described in [4.26] and basic linear algebra [4.27].

In the remainder of this section, we consider the following general problem.

Given: an observation model

$$y = f(x, \eta). \quad (4.2)$$

Estimate: $x \in \text{Re}(n)$ from observations $y \in \text{Re}(m)$ where η is an unknown disturbance taking values in $\text{Re}(k)$ and f is a known mapping from $\text{Re}(k+n)$ to $\text{Re}(m)$.

We divide our discussion into two topical areas:

- methods for performing estimation on batch and sequential data when f is linear
- methods for performing estimation on sequential data when f is nonlinear

Estimation Techniques for Batch and Sequential Data with Linear Models

In this section, we discuss linear and linearized estimation techniques for sequential data, including the Kalman filter and extensions thereof. Our goal is to provide an overview of techniques available. The reader may also wish to consult more in-depth references such as [4.16, 28, 29] and (Chap. 25) for additional information.

We first consider the case when f in (4.2) is linear in its arguments. In this case, we can write

$$y = Fx + B\eta, \quad (4.3)$$

where $F \in \text{Re}(m \times n)$ defines the (linear) relationship between the unknown x and the observation y and $B \in \text{Re}(m \times m)$. For the moment, we will drop B and assume that η represents the complete disturbance model of the system.

The *least-squares* method of estimating x from y proceeds by solving the optimization problem

$$\min_x \|Fx - y\|^2. \quad (4.4)$$

This optimization has a unique solution \hat{x} if and only if the matrix F has full column rank. In this case, the

solution can be computed by solving the following linear system:

$$F^\top F \hat{x} = F^\top y. \quad (4.5)$$

In some cases, there may be reason to believe that some observed elements are more reliable than others, and hence should contribute more to the final estimate. This information can be incorporated by modifying (4.4) to include a diagonal positive-definite weighting matrix W as

$$\min_x (Fx - y)^\top W (Fx - y). \quad (4.6)$$

The solution is then given by solving:

$$(F^\top W F) \hat{x} = F^\top W y. \quad (4.7)$$

Although (4.3) included a disturbance component (in the form of η), the parameter estimates computed in (4.5) or (4.7) made no explicit use of this quantity. However, we can often model the noise characteristics of the underlying sensor using a statistical model and recast our original estimation problem to incorporate this information. One common method is to compute the *maximum-likelihood estimate* (MLE), which is a value \hat{x} such that

$$p(y|\hat{x}) = \max_x p(y|x). \quad (4.8)$$

For the linear additive model of (4.3), the likelihood function can be expressed in a particularly simple form. Suppose that η is described by a fixed, known probability density function D . The likelihood function is then given by

$$p(y|x) = D(y - Fx). \quad (4.9)$$

The MLE can be related to the previous least-squares method as follows. Suppose that $\eta \sim N(0, \Lambda)$, where N denotes a multivariate Gaussian density function with (mean) 0 and covariance Λ . Upon observing that the maximizing the value of the likelihood function is equivalent to minimizing the negative log of the likelihood function, a short series of calculations shows that the optimal maximum-likelihood estimate is computed by weighted least squares with $W = \Lambda^{-1}$.

Finally, there is often a reason to include the idea that some parameters are more likely a priori to occur as others. For example, when observing a car driving on an expressway, a velocity of 60 mph is much more likely than either 20 mph or 300 mph. This information can be captured in *prior statistics* on the unknown value x .

Given a prior probability density on x , $p(x)$, Bayes theorem states that

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\int p(y|x)p(x)dx}. \quad (4.10)$$

The *maximum a posteriori probability* (MAP) estimate is the value \hat{x} such that

$$p(\hat{x}|y) = \max_x p(x|y). \quad (4.11)$$

In general, the solution to this optimization problem can be quite complex. Rather than pursue this course further, we consider another alternative. Namely, provided the second moments of $p(x|y)$ exist, it is possible to produce a *least-squares* estimate, in a statistical sense, by solving the following optimization problem over an unknown function δ :

$$\min_{\delta} E \|\delta(y) - x\|^2. \quad (4.12)$$

That is, the best function δ is one that produces an estimate of x from y with *minimum mean-square error* (MMSE). Thus, the estimator δ is often referred to as a MMSE estimator.

It can be shown that, in the general case, the *optimal* decision rule δ^* is the conditional mean [4.20]

$$\delta^*(y) = E[x|y]. \quad (4.13)$$

Unfortunately, this expression, as with the MAP estimate defined above, can be extremely difficult to compute for the general case. Later we consider methods for computing approximations to (4.13). For now, we again consider our previous linear observation model (4.3) (without B). Additionally, we suppose that x and η are *independent* random variables with finite second moments, and both are *zero-mean* random variables. Note that the latter is not really a restriction since it can be accomplished by simply defining a new variable $x' = x - E[x]$. Finally, we will consider only *linear* functions δ , that is, we can write $\hat{x} = \delta(y) = Ky$.

With this, (4.12) can be expanded as

$$\begin{aligned} E \|\delta(y) - x\|^2 &= E \|Ky - x\|^2 \\ &= E \|K(Fx + \eta) - x\|^2 \\ &= E \|(KF - I)x\|^2 + E \|K\eta\|^2 \\ &= \text{tr} \left[(KF - I)\Lambda(KF - I)^\top + K\Sigma K^\top \right]. \end{aligned} \quad (4.14)$$

Here, the independence of x and η and the fact that they are both zero mean has eliminated several terms. The final step makes use of the fact that $\|x\|^2 = \text{tr}(xx^\top)$.

Taking derivatives with respect to K and setting them equal to zero yields the solution

$$K = \Lambda F^\top (F \Lambda F^\top + \Sigma)^{-1}. \quad (4.15)$$

Thus, in this case the optimal estimate is given by a linear function of the observation, where the linear term depends only on the variance of the underlying random variables and the linear term defining the observation system.

If x is not zero-mean, but has mean μ , it is not hard to show that the optimal estimate is

$$\hat{x} = Ky + (I - KF)\mu, \quad (4.16)$$

and that the variance of the estimate Λ^+ is

$$\Lambda^+ = (I - KF) \Lambda. \quad (4.17)$$

The interested reader may wish to work this out for a few simple cases, for example, if $\Lambda = \Sigma$ and $F = I$, $K = 1/2I$ and thus $\hat{x} = y + \mu$ – a simple average – with variance $\Lambda^+ = 1/2\Lambda$.

When both the observation noise and prior statistics are Gaussian distributions, then it can be shown that the solution we have derived is also the MAP estimate for the unknown x [4.20].

The Kalman Filter. With this as background, we are now in a position to define the discrete-time Kalman–Bucy filter [4.30] for linear systems. Consider the following time-series model:

$$x_{t+1} = Gx_t + w_t, \quad (4.18)$$

$$y_t = Fx_t + \eta_t, \quad (4.19)$$

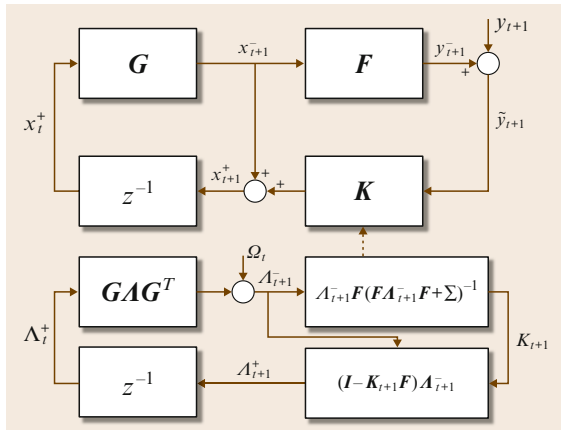


Fig. 4.11 A summary of the Kalman filter

where G is an $n \times n$ matrix describing the system time evolution and x_0 is distributed according to a Gaussian distribution with mean \hat{x}_0 and variance Λ_0 . In addition w_t and η_t are zero-mean Gaussian independent random variables for all t , w_t is independent of $w_{t'}$ for all $t \neq t'$, and likewise η_t is independent of $\eta_{t'}$ for all $t \neq t'$. Finally, η_t has variance Σ_t and w_t has variance Ω_t .

Given an observation y_1 it is possible, using the derivation of the previous section, to compute an updated estimate \hat{x}_1 with variance Λ_1 . Note, that the solution is a linear combination of two Gaussian random variables: the observation value y_1 and the prior estimate \hat{x}_0 . As any linear combination of Gaussian random variables is also a Gaussian random variable, it follows that the updated estimate is also Gaussian.

Now, we add one additional step: projection through the dynamic model. To describe this, superscripts minus and plus will denote before and after the estimation step, respectively. Thus, given an estimate \hat{x}_t^+ with variance Λ_t^+ , the projection ahead one time step produces

$$\hat{x}_{t+1}^- = G\hat{x}_t^+, \quad (4.20)$$

$$\Lambda_{t+1}^- = G\Lambda_t^+G^\top + \Omega_t. \quad (4.21)$$

At this point, a new observation y_{t+1} is acquired and the cycle repeats. The summarization of the complete Kalman filtering algorithm for linear systems is shown in Fig. 4.11.

It is possible to show that the Kalman filter is the *optimal* filter, under the stated assumptions, in the mean-square sense. It is also the optimal *linear* filter when either or both Gaussian assumptions do not hold.

Nonlinear Estimation Techniques for Sequential Data. The results of the previous subsection presume a linear form for the relationship between the observation and system state, additive noise, and a linear relationship describing the state evolution. Furthermore, the stated results are globally optimal for systems with Gaussian observation and driving noise, but are only the best *linear* estimator if the noise sources are non-Gaussian.

As noted at the outset, the more general nonlinear (discrete-time) system description is

$$\begin{aligned} x_{t+1} &= g_t(x_t) + w_t, \\ y_t &= f_t(x_t) + \eta_t, \end{aligned} \quad (4.22)$$

where, for the moment, the noise model continues to be additive.

Although this model contains nonlinear elements, it is still possible to apply a variant of the Kalman filter,

the extended Kalman filter (EKF) by making use of the Taylor-series expansion of the nonlinear elements about the current estimates. Let J_f (resp. J_g) denote the Jacobian matrix of the function f (resp. g). Supposing that an estimate at time step $t-1$ exists, the first-order expansion of (4.22) about this point yields

$$x_{t+1} = g_t(\hat{x}_{t-1}) + J_{g_t}(\hat{x}_{t-1})(x_t - \hat{x}_{t-1}) + w_t, \quad (4.23)$$

$$y_t = f_t(\hat{x}_{t-1}) + J_{f_t}(x_t - \hat{x}_{t-1}) + \eta_t. \quad (4.24)$$

Rearranging yields a linear form appropriate for the previously defined Kalman filter

$$\tilde{x}_{t+1} = x_{t+1} - g_t(\hat{x}_{t-1}) + J_{g_t}\hat{x}_{t-1} = J_{g_t}x_t + w_t, \quad (4.25)$$

$$\tilde{y}_t = y_t - f_t(\hat{x}_{t-1}) + J_{f_t}\hat{x}_{t-1} = J_{f_t}x_t + \eta_t. \quad (4.26)$$

In this form, \tilde{x} and \tilde{y} are new *synthetic* state and observation variables, J_{g_t} plays the role of G , and J_{f_t} plays the role of F .

It is worth noting that the EKF iterations are essentially a form of weighted Newton iterations (i. e., an iterative nonlinear estimation method). As a result, it is often useful to iterate more than once *on the same observation* while holding the variance terms fixed. This allows the estimator to converge to a solution in the presence of large disturbances or significant nonlinearities. Only after convergence are the variance terms updated. This version of the Kalman filter is referred to as the *iterated extended Kalman filter* (IEKF).

4.3.2 Other Approaches to Estimation

In the previous section, we reviewed the most common and widely used estimation methods. However, there are several alternative methodologies for solving parameter estimation problems. Here we briefly introduce two: sequential importance sampling and graphical models.

Sequential Importance Sampling

Much of the discussion heretofore has centered around the notion of approximating everything known about the system state using an estimated mean and covariance. An alternative presents itself by simply going back to Bayes theorem which states, in general, that

$$p(x_n | y_1, y_2, \dots, y_n) = \frac{p(y_1, y_2, \dots, y_n | x_n) p(x_n)}{p(y_1, y_2, \dots, y_n)}. \quad (4.27)$$

Assuming that y_n is independent of all prior observations and states given x_n , and that x_n is independent of x_{n-k}

for $k > 1$ given x_{n-1} , this expression simplifies to

$$p(x_n | x_{n-1}, y_n) = \frac{p(y_n | x_n) p(x_n | x_{n-1})}{p(y_n | x_{n-1})}. \quad (4.28)$$

Recall that the optimal mean-square estimate is given by the conditional mean which, in this case, is

$$\delta^*(y_n) = E[x_n | y_n]. \quad (4.29)$$

In fact, we essentially showed that the Kalman filter is a special case of this result for linear systems corrupted by Gaussian noise.

The difficulty in implementing this procedure in the general case ultimately comes down to the problem of representing and computing with the distributions that arise in the nonlinear, non-Gaussian case. However, suppose that the heretofore continuous variable x_n only took on a discrete set of values. In this case, computing Bayes theorem and other associated statistical quantities reduces to a straightforward set of computations on this discrete set of variables. This can be simply done for any distribution and any set of transformations.

Sequential important sampling (also known as particle filtering, condensation, and a variety of other names) is a way of taking this approach in a statistically sound manner. In order to perform sequential importance sampling, it is assumed that

1. it is possible to *sample from* the likelihood function $P(y_n | x_n)$, and
2. it is possible to *sample from* the dynamical model $P(x_n | x_{n-1})$.

Note the emphasis on sampling – there is no need to explicitly exhibit an analytical form of the likelihood function or of the dynamical model.

Given this, sequential important sampling, in its simplest form, can be written as follows.

1. Let $\pi_{n-1} = \{\langle x_{n-1}^k, w_{n-1}^k \rangle, k = 1, 2, \dots, N\}$ represent a set of sample points x_{n-1}^k together with a set of weights w_{n-1}^k with $\sum w_{n-1}^k = 1$.
2. Compute a new set of N samples $\pi_{n-1}^- = \{\langle x_n^k, 1/N \rangle, k = 1, 2, \dots, N\}$ as
 - a) choose a sample point x_{n-1}^k with probability proportional to its weight w_{n-1}^k ;
 - b) Sample from $P(x_n | x_{n-1}^k)$ given x_{n-1}^k with weight $1/N$;
3. Compute $\pi_n = \{\langle x_n^k, P(y_n | x_n^k) \rangle, k = 1, 2, \dots, N\}$.

It is easy to see that this set of steps is now in the form of a recursive filter. Furthermore, at any time any statistic

of the associated distribution can be approximated from the set of samples and associated weights.

Sampling-based filters of this form have found wide applicability in a variety of challenging areas where linear estimation techniques do not suffice. These techniques have been particularly successful, for problems with low state dimension (typically $n \leq 3$) and well-constrained dynamics. For higher-dimensional problems or systems exhibiting high dynamic variability, the number of particles necessary to obtain good approximations can become prohibitively large. However, even in these cases, sampling-based systems can sometimes be engineered to produce acceptably good results.

Graphical Models

Graphical models are a class of models that represent dependence and independence relationships among a set of variables. Common examples of graphical models include Bayes nets, influence diagrams, and neural nets. Here, we focus on Bayes nets as a specific example.

A Bayesian network is a directed acyclic graph consisting of nodes representing random variables, and directed arcs representing probabilistic relationships between pairs of random variables. Let $\text{parents}(X)$ denote the set of nodes which have arcs terminating at X , and let X_1, X_2, \dots, X_N be the N random variables in the graph. Then we can write

$$P(X_1, X_2, \dots, X_N) = \prod_{i=1}^N P(X_i \mid \text{parents}(X_i)) \quad (4.30)$$

For example, a Bayesian network representing a mobile robot performing localization is shown in Fig. 4.12. This graphical model encodes the sequential form of the problem and is thus an example of a so-called *recurrent network*. More discussion of such models can be found in [4.31].

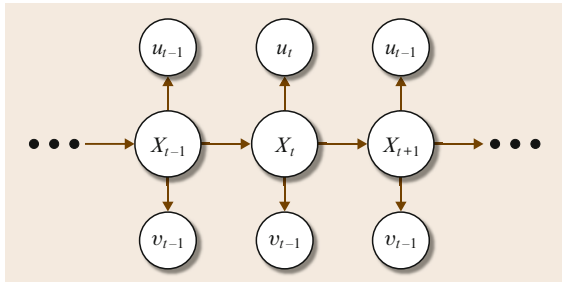


Fig. 4.12 An example of robot localization expressed as a graphical model

The structure of a Bayesian network encodes various independence relationships among variables. By exploiting these independence relationships, it is possible to design efficient inference algorithms. In particular, graphs which are acyclic even in their undirected form (referred to as *polytrees*) admit linear inference algorithms. More general graphs can be solved using various types of iterative methods. In particular, if the distributions in the network are of a continuous type, variations on sequential importance sampling can be used to solve problems in an approximate sense [4.32].

4.3.3 Robust Estimation Methods

In our previous discussions, we generally assumed that all of the data was *good*, meaning that it was perhaps corrupted by noise but ultimately carried information about the problem at hand. However, in many cases, the data may contain so-called *outliers* – data points that are either much more highly corrupted than *typical* data, or which are completely spurious. For example, in our mapping application we might occasionally obtain range data through multiple reflections. Thus, while scanning a straight wall, most of the points would lie on a straight line, but occasionally we would have a data point that has a completely inconsistent range value.

The problem is that many common estimation methods are quite sensitive to data outliers. Consider a very simple case: estimating a single scalar value x by averaging a series of observations X_1, X_2, \dots, X_N . Then we can write our estimate \hat{x} as

$$\hat{x} = \sum_{i=1}^N X_i / N. \quad (4.31)$$

Now, without loss of generality, suppose that X_N is an outlier. We can rewrite the above as

$$\hat{x} = \sum_{i=1}^{N-1} X_i / n + X_N / n. \quad (4.32)$$

It is now easy to see that we can produce *any* value of \hat{x} by manipulating X_n . In short, a single outlier can create an arbitrarily poor estimate. More generally, the solution to any least-squares problem, e.g., estimating a line from laser range data, takes the general form $\hat{x} = My$. By the same argument as above, it is easy to show that any least-squares solution is likewise susceptible to outliers.

The field of *robust statistics* studies the problem of estimation or decision making when the underlying data are contaminated by outliers. In robust statistics, there

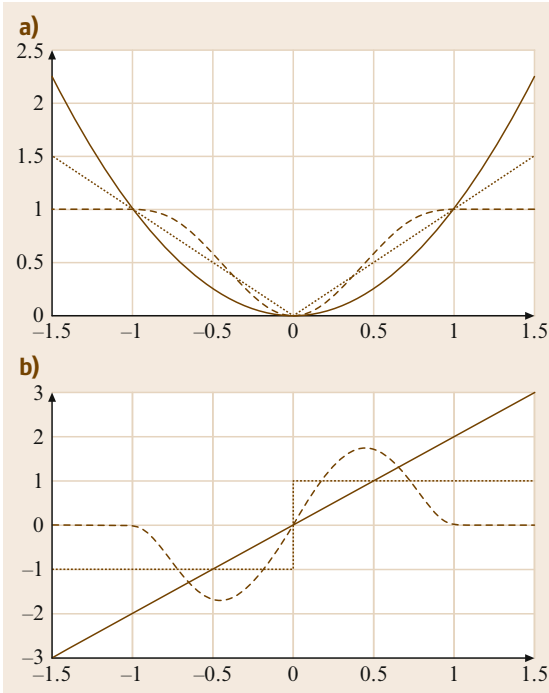


Fig. 4.13 (a) Three common robust M-estimation functions: the square function, the absolute value, and the Tukey biweight function. (b) The corresponding influence functions

are two important concepts: the *breakdown point* and *influence function*. The breakdown point is the proportion of outliers (i. e., data with arbitrarily large errors) that an estimator can tolerate before producing arbitrarily large errors in an estimate. We argued above that least-squares methods have a breakdown point of 0% since the estimate can be perturbed arbitrarily far by a single observation. By comparison, we might compute an estimate by taking the median of the data, which has a breakdown point of 50% – up to half of the data can be outliers and meaningful results may still be produced.

Whereas the breakdown point quantifies *how many* outliers can be tolerated, the influence function quantifies *how much* an outlier affects an estimate. In the case of least squares, the influence function is linear. One way of creating new estimators with better robustness is the method of *M-estimators* [4.19]. To produce an M-estimate, we consider the following minimization problem:

$$\min_{\hat{x}} \sum_{i=1}^N \rho(\hat{x}, y_i). \quad (4.33)$$

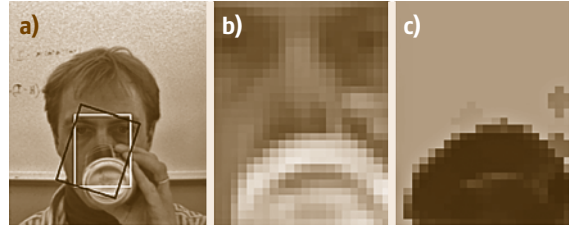


Fig. 4.14a–c An example of using an M-estimate implemented via IRLS for visual tracking (after [4.33]). (a) Results of a face tracker in a single frame of video. The black frame corresponds to a tracking algorithm without outlier rejection and the white frame corresponds to the algorithm with outlier rejection. (b) Magnified view of the region in the white frame; (c) the corresponding weighting matrix in which darker areas mark outliers

Note that defining $\rho(a, b) = (a - b)^2$ leads to a least-squares solution. However, we can now choose other functions with better resistance to outliers. Figure 4.13 shows three common examples.

Note that, in general, the optimization of (4.33) is nonlinear and the result will often not exist in closed form. Interestingly, it is often possible to solve this problem using the method of *iteratively reweighted least squares* (IRLS) [4.28, 34]. The idea behind IRLS is quite simple. Recall that in (4.7) we introduced a weighting matrix W . Suppose that, through some means, we knew which data points were outliers. In this case, we could simply set the weights for those points to zero, and the result would be the least-squares estimate on the remaining (good) data.

In IRLS, we alternate between hypothesizing outliers (through reweighting) and solving to produce a solution (through least squares). Typically, the weight for a point depends on the residual error of the estimate. That is, suppose we compute

$$r = y - F\hat{x}. \quad (4.34)$$

Let $\psi(y) = d\rho/dx|_{\hat{x}}$; then we can set $W_{i,i} = \psi(y)/r_i$. It can be shown that in many cases this form of weighting will lead to convergence. An example of using IRLS techniques for video tracking is shown in Fig. 4.14.

Voting-Based Methods

Another common method for dealing with outliers is to choose a set of data and let it *vote* for a result. We discuss two common methods: **RANSAC** [4.9] and least median of squares (LMedS) [4.35].

In both cases, we start with the idea that, amidst all of the data (including outliers), there is an estimate

that is consistent with the *good* data. The problem is to choose that estimate. Consider, however, our problem of estimating a line from laser data, and suppose we have 100 laser points. All we really need is to choose two points correctly, fit a line, and then count how many other points are consistent with this line. If we (conservatively) estimate that 3/4 of the data is good, then the odds of choosing two *good* points is 9/16, or equivalently, the odds of one or both points being outliers is 7/16. If we now repeat this process a few (e.g., ten) times, then the odds that *all* of our choices are bad is $(7/16)^{10} = 0.025\%$. To put it in other terms, there is a 99.975% chance we have chosen a *good* pair of points.

How do we decide to accept a sample? In **RANSAC**, we *vote* by counting the number of samples that are consistent with an estimate to within a given distance threshold. For example, we would choose points that are within a fixed distance to the line we estimated. We choose the candidate estimate with the largest number of votes. In LMedS, we instead compute the median distance of all of the samples to the line. We then choose the estimate with the least median value.

It is not hard to see that LMedS has a breakdown point of 50% of the data. **RANSAC**, on the other hand, can have a breakdown point that is potentially larger, but it requires the choice of a threshold. **RANSAC** also has the advantage that, once the inliers are identified, it is possible to compute a least-squares estimate from them, thus reducing the noise in the estimate.

Both **RANSAC** and LMedS can also provide good starting solutions for a robust iterative method such as IRLS.

4.3.4 Data Association Techniques

The previous section considered the case where there is a *known* relationship between observations and a quantity to be estimated. However, as was illustrated in our initial mobile robot mapping problem, it may be the case that we also have to compute this correspondence in conjunction with estimation. In this case, an essential step in estimation is the *data association* problem: producing a correspondence between the observed data and quantities to be estimated.

The literature on this problem is enormous; here we will focus on a few specific methods that have found wide use. We will also separate our discussion into *causal* (or sequential) association methods commonly used when filtering time-series data and *noncausal* (or batch) methods that can be used when the complete data

set is available for processing. The latter is typically treated with methods for data *clustering*.

In both cases, we can extend our previous models and notation to include uncertainty as to the underlying source of the data. To this end, we will use a superscript on quantities to denote the observation model. Thus, our observation model becomes

$$x_{t+1}^k = g^k(x_t^k) + w_t^k, \quad (4.35)$$

$$y_t^k = f_t^k(x_t^k) + \eta_t^k, \quad (4.36)$$

where $k = 1 \dots M$.

Clustering on Batch Data

Following the same course as our previous discussion on point estimation, let us first consider the case where we do not make any statistical assumptions about the data, and we have no system dynamics. Thus, we are simply given the observations y_1, y_2, \dots, y_M . We have unknown underlying parameters x^1, x^2, \dots, x^N (for the moment, we take N as known). Our goal is to compute an *association mapping* π such that $\pi(j) = k$ if and only if y_j arose from the model parameters x^k .

k-means Clustering

The *k*-means algorithm for clustering and data association is simple, well established, and forms a good starting point for our discussion. Here, we assume that $f(x) = x$ – that is, we are provided with noisy observations of the underlying state vectors. The *k*-means algorithm then proceeds as follows.

1. Pick N cluster centers $\{\hat{x}^i\}$.
2. For each observation y_j , associate it with the closest cluster center, that is, set $\pi(j) = i$, where

$$d(\hat{x}^i, y_j) = \min_k d(\hat{x}^k, y_j) \quad (4.37)$$

for some distance function d (typically the Euclidean distance).

3. Estimate the mean of the observation associated with each cluster center as

$$\hat{x}^i = \sum_{j, \pi(j)=i} y_j. \quad (4.38)$$

4. Repeat steps 2 and 3.

In many cases and with good initialization, *k*-means works quite well. However, it can also fail to produce good clusters, and there is no guarantee that it will even converge to a solution. It is common to repeat the algorithm several times from different initial conditions and take the result that has the *best* outcome. Note

also that the extension to linear observation models is straightforward by including F in (4.3) by defining

$$d(\hat{x}^i, y_j) = \|F\hat{x}^i - y_j\| \quad (4.39)$$

and replacing (4.38) with the corresponding least-squares estimator. Going a step further, if we have a statistical model for observed data, then we could make use of the likelihood function introduced earlier and define $d(\hat{x}^i, y_j) = p(y_j|\hat{x}^i)$ and make use of the MLE in (4.38).

One disadvantage of the k -means algorithm is that, even when we have known statistical models, it is not guaranteed to converge. However, a variation, known as *expectation maximization*, can be shown to converge.

Expectation Maximization for Data Association and Modeling

The expectation-maximization (EM) algorithm [4.36] is a general statistical technique for dealing with missing data. In previous discussion, we made use of maximum-likelihood estimation to maximize the conditional probability of observed data given a set of unknown parameters. However, our use of MLE presumed that we had complete knowledge of the data. In particular, we knew the association between the data elements and models.

Let us now assume that some of our data is missing. To this end, define \mathcal{Y}_O and \mathcal{Y}_U as the *observed* and *unobserved* data, respectively. We then note that we can write

$$p(\mathcal{Y}_O, \mathcal{Y}_U|x) = p(\mathcal{Y}_U|\mathcal{Y}_O, x)p(\mathcal{Y}_O|x). \quad (4.40)$$

Suppose now that we make a guess for \hat{x} , and we have a distribution over the unknown data \mathcal{Y}_U (where this comes from we will discuss in a minute). It follows that we could compute the *expected value* of the log-likelihood function (recall that maximizing the log likelihood is equivalent to maximizing the likelihood) as

$$Q(x, \hat{x}) = E_{\mathcal{Y}_U} [\log p(\mathcal{Y}_O, \mathcal{Y}_U|x) | \mathcal{Y}_O, \hat{x}]. \quad (4.41)$$

Note that we differentiate between the fixed value \hat{x} that is usually needed to define the distribution over the unknown data and the unknown x of the log-likelihood function.

Ideally, we would then like to choose values for x that make Q large. Thus, we can choose a new value according to the iterative rule

$$\hat{x}_i = \arg \max Q(x, \hat{x}_{i-1}). \quad (4.42)$$

What can be shown is that this iteration will converge to some local maximum of the objective function Q . It is

important to note that there is no guarantee that this is, however, the *global* maximum.

How do we connect this with clustering? We consider the observed data to be just that, the data we have observed. Let the unobserved data be the *association values* $\pi(j)$, $j = 1, 2, \dots, M$ that determine which model the observed data items originate from. Note that this is a discrete random variable. Let us further assume that N underlying clusters are distributed according to a Gaussian distribution with mean x_i and covariance Λ_i . Let the unconditional probability that a particular data item y_j comes from cluster i be α_i . The unknown parameters are then $\theta = \{x_1, x_2, \dots, x_N, \Lambda_1, \Lambda_2, \dots, \Lambda_N, \alpha_1, \alpha_2, \dots, \alpha_N\}$. We now use $-$ and $+$ to denote prior and updated parameter estimates, respectively. For conciseness, we also define $w_{i,j} = p(\pi_j = i | y_j, \theta)$ and we use a subscript plus to denote updated parameter estimates. Then, after a series of calculations [4.36], the EM algorithm for data clustering becomes:

E-Step:

$$w_{i,j} = \frac{p(y_j | \pi(j) = i, \theta) \alpha_i}{\sum_i p(y_j | \pi(j) = i, \theta) \alpha_i}. \quad (4.43)$$

M-Step:

$$\hat{x}_i^+ = \sum_j y_j w_{i,j} / \sum_j w_{i,j}, \quad (4.44)$$

$$\Lambda_i^+ = \sum_j y_j (y_j)^t w_{i,j} / \sum_j w_{i,j}, \quad (4.45)$$

$$\alpha_i^+ = \sum_j w_{i,j} / \sum_i \sum_j w_{i,j}. \quad (4.46)$$

From this, we can see that EM produces a type of soft clustering, as opposed to k -means which produces specific decisions (in terms of $w_{i,j}$) as to which cluster an observation belongs. In fact, the result of estimation is the maximum-likelihood estimate of a *Gaussian mixture model* which has the form

$$p(y|\theta) = \sum_j \alpha_j N(y|\hat{x}_j, \Lambda_j), \quad (4.47)$$

where $N(\cdot)$ denotes a Gaussian density function. Figure 4.15 shows the results of executing the EM algorithm on data sampled from a Gaussian mixture model.

Recursive Filtering

In the batch methods described above, we do not have a priori information on state parameters. In the case of recursive filtering, we have the advantage that prior state

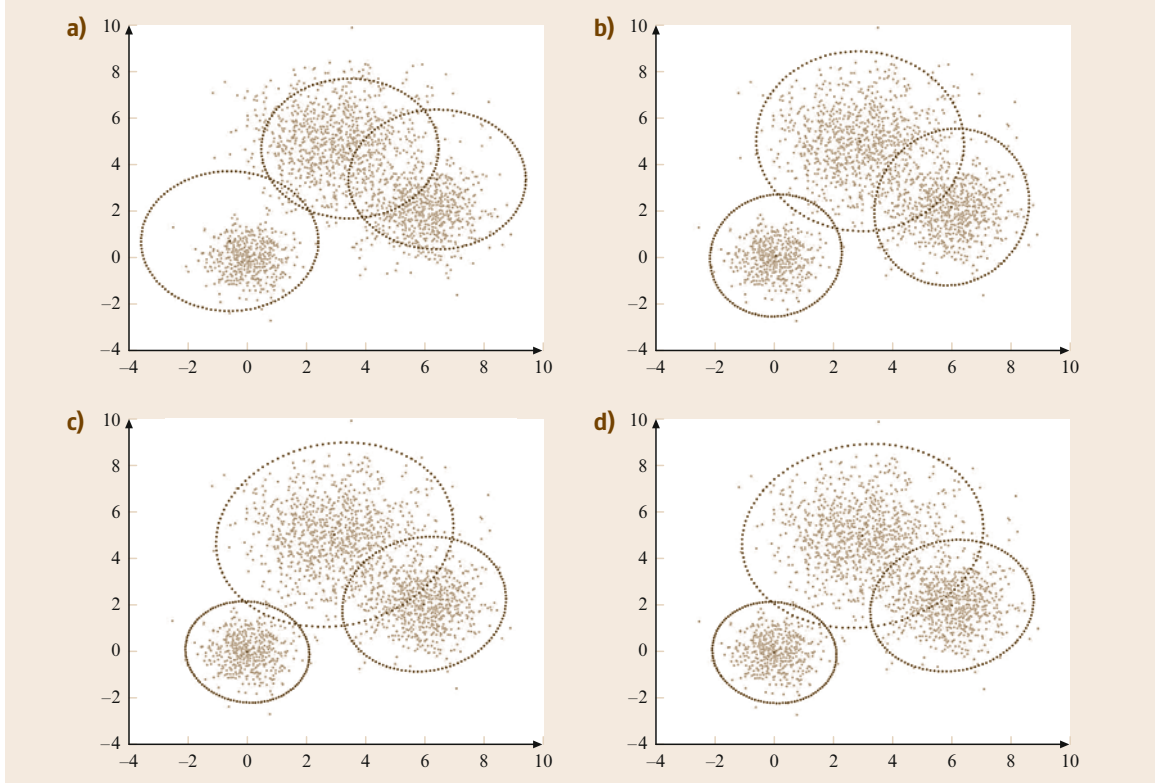


Fig. 4.15a–d An example of clustering using expectation maximization. The figures are the results at iterations 1, 2, 5, and 10

estimates, \hat{x}_t^k and Λ_t^k , are available for processing at time $t + 1$. As before, for data y_t^i , $i = 1 \dots N$, the problem is to determine a mapping $\pi : \{1 \dots N\} \rightarrow \{1 \dots M\}$ which associates data element i to model $k = \pi(i)$. In some cases, it is also useful to include an *outlier process* to handle data that comes from no known model. For this purpose, we can include 0 in the range of the function, and use the mapping to zero as an outlier.

Nearest-Neighbor Association

Analogous to k -mean clustering, a simple way of producing a data association is to compute the data association value as

$$\pi(i) = \arg \min_j d(F^j \hat{x}^j, \hat{y}^i). \quad (4.48)$$

However, nearest-neighbor methods do not take into account what we know about either the sensor data or the estimate. That is, we may have a very very good estimate of some model i and a very very bad estimate for some other model j . If a sensor observation is equidistance

between them, does it make sense to flip a coin? Odds are that it is more likely to come from j (with a larger variance) than i (with a smaller variance).

A commonly used measure that can take this into account is the *Mahalanobis* distance [4.37]. The idea is to weight each value by its variance as

$$m(y_1, y_2) = (y_1 - y_2)(\Lambda_1 + \Lambda_2)^{-1}(y_1 - y_2)^\top. \quad (4.49)$$

Thus, distances are scaled inversely with uncertainty. In the case above, the observation with a higher variance would produce the smaller distance, as desired.

Even using this as a weighting method, it is still possible that we will make an error in data association. From an estimation point of view, this will introduce an outlier in the estimation process with, as discussed above, potentially disastrous results. Another approach, analogous to IRLS, is instead to weight the data based on the distance to a model. This leads naturally to the notion of a *data association filter*. We refer the reader to [4.18] for extensive discussions of these techniques.

4.3.5 Modeling Sensors

To this point, we have introduced several sensing modalities, and we have discussed several methods for estimation. However, the latter often rely on having statistical models for the former. Thus, no chapter on sensing and estimation would be complete without a short discussion of modeling sensors.

Developing a sensor model potentially involves four major elements: (1) creating a physical model, (2) determining a sensor calibration (3) determining an error model, and (4) identifying failure conditions.

The physical model is the relationship f between the underlying quantities of interest (x) and the available data (y). In many cases, this relationship is obvious, e.g., the distance from a laser sensor to a surface in the world. In others, it may be less so, e.g., what is the right model relating intensities in multiple camera images to the distance to an observed point? In some cases, it may be necessary to include computational processes, e.g., feature detection and correspondence, in the sensor model.

Once a physical model is determined, there is often a process of sensor calibration. Such procedures are typically specific to the sensor in question, for example, the imaging geometry of a perspective camera system requires identification of two scale parameters (governing image scale) and the location of the optical center (two additional parameters). There are also often lens distortion parameters. These parameters can only be determined by a careful calibration procedure [4.5].

Once a calibrated physical sensor model is available, determining an error model typically involves performing an identification of the statistical parameters. Ideally, the first step is to determine an empirical distribution on errors. However, this can often be difficult, as it requires knowing accurate *ground truth* for the underlying unknown parameters. This often requires the development of a laboratory setup that can simulate the expected sensing situation.

Given such an empirical distribution, there are several important questions, including: (1) Are observations statistically independent? (2) Is the error distribution unimodal? and (3) Can the essential aspects of the

empirical error be captured using common statistical quantities such as the data variance? We refer the reader to books on statistics and data modeling [4.38] for further information on this topic.

Finally, it is important to understand when sensors can and cannot provide reliable data, for example, a laser sensor may be less accurate on dark surfaces than on light ones, cameras do not produce meaningful data if the lighting is too bright or too dark, and so forth. In some cases, there are simple clues to these conditions (e.g., simply looking at the intensity histogram of a camera image can quickly determine if conditions are suitable for processing). In some cases it is only possible to detect conditions in context (e.g., two range sensors disagree on the distance to a surface). In some cases failure is only detectable in retrospect, e.g., after a 3-D surface model is built it is apparent that a hypothesized surface would be occluded by another and must therefore be a multiple reflection. In a truly robust sensing system, all available possibilities for verifying sensor operation should be exploited.

4.3.6 Other Uncertainty Management Methods

Due to the limitations of space, we have necessarily limited our discussion to cover the most commonly used sensing and estimation methods. It is important to note that many other alternative uncertainty management methods have been proposed and employed with success.

For example, if it is known that sensing error is bounded, constraint-based methods can be quite effective at performing point estimation [4.39, 40]. Alternatively, if only partial probability models can be identified, Dempster–Shafer methods can be employed to make judgments [4.41].

Fuzzy logic allows graded membership of a set. With fuzzy set theory it is possible to have partial membership. As an example in classification of data it might be difficult to select between two categories such as *average* and *tall* and gradual shifts may make sense. Such methods have for example been used for situation assessment and navigation as reported by [4.42] for the DAMN architecture.

4.4 Representations

Sensor data can be used directly for control but it is also used for estimation of the state of the robot and/or the world. The definition of *state* and the appropriate methods for estimation are closely related to the representation adopted for the application.

There are a rich variety of possible world representations including most typical geometric elements such as points, curves, surfaces, and volumes. A fundamental aspect in robotics is the concept of rigid-body *pose*. The pose of a robot or an entity in the world is characterized by position and orientation with respect to a reference frame.

In general, pose is represented by the pair (\mathbf{R}, \mathbf{T}) . Here \mathbf{R} is the orientation of the object represented by a rotation matrix with respect to a reference frame. Similarly, \mathbf{T} represents the translation of the object with respect the reference frame. There is a rich set of potential representations for the transformation between reference frames as detailed in the chapter on Kinematics (Chap. 1) and in [4.43].

Sensory data is acquired in a local sensor reference frame, for example, a sonar transducer, a laser scanner, and a stereo imaging system would all measure distances to surfaces in the world relative to their own frame. However, if the goal is to combine this information into a common world model, the data must be transformed into a robot-centered reference frame, or possibly into a fixed world (inertial) reference frame. In particular, the world-centered reference frame enable simple transfer across robot motions and communication to other robots and/or users.

For the purposes of discussion, most representations for the integration of sensor data can be categorized into four general classes of models:

- raw sensor data models
- grid-based models
- feature-based models
- symbolic or graphical models

Naturally, it is also possible to combine elements of these four categories to achieve hybrid models of the environment.

4.4.1 Raw Sensor Representations

For simple feedback control [4.44] it is common to integrate raw sensory data directly into the control system, as in many cases it is unnecessary to have a *world model* for the control. For example, proprioceptive sensing is often

used in this manner: basic trajectory control makes direct use of encoder information from joints, and force control operates directly from force or torque information from force sensors.

Raw sensor models are less common with exteroceptive sensing, but there are cases where it can be useful. One example is mobile robot mapping from dense point data. This approach has in particular been made popular for laser range sensors, where scan alignment is used for the generation of point-based world models. The work by [4.45, 46] demonstrates how a number of laser range scans can be combined into a joint model of the environment. More formally a scan of the environment at time t is represented as a point set

$$\mathcal{P}_t = \{p_i = (\rho_i, \theta_i) | i \in 1 \dots N\}. \quad (4.50)$$

Two different scans \mathcal{P}_t and \mathcal{P}_{t+1} are then aligned through a standard SE(3) transformation. The estimation of the transformation is typically achieved through use of the ICP algorithm [4.13]., assume that $T^{[0]}$ is an initial estimate of the transformation between the two point sets and that $\|p_i - p_{t+1}\|$ is the Euclidean distance between a point from \mathcal{P}_t and a point from \mathcal{P}_{t+1} . If furthermore \mathbf{CP} is a function to locate the closest point from one set in the other set, then let \mathcal{C} be the set of point correspondences between the two sets. Through iterations of the following algorithm:

1. compute $\mathcal{C}_k = \bigcup_{i=1}^N \{p_i, \mathbf{CP}[T^{[k-1]}(p_i, \mathcal{P}_{t+1})]\}$,
2. estimate the $T^{[k]}$ that minimizes the LSQ error between the points in \mathcal{C}_k until the error has converged

an estimate of the scan alignment can be found and a joint model of the environment can be constructed.

The model is simple to construct and well suited for integration of sensor data from a single modality. Typically the model does not include information about uncertainty and, as the model grows the complexity, $O(\sum_i |\mathcal{P}_i|)$ becomes an issue.

4.4.2 Grid-Based Representations

In a grid-based representation the world is tessellated into a number cells. The cells can contain information about environmental features such as temperature, obstacles, force distribution, etc. The dimensionality of the grid is typically two or three, depending on the application. The tessellation can either be uniform or tree based using quad-tree or oct-trees [4.47]. The tree-based methods are in particular well suited for handling of in-

homogeneous and large-scale data sets. In a grid model each cell contains a probability over the parameter set. As an example, when using the grid model for representation of a physical environment, the cell specifies *occupied* or *free* and the cell encodes the probability $P(\text{occupied})$. Initially where there is no information the grid is initialized to $P(\text{occupied}) = 0.5$ to indicate unknown. It is further assumed that sensor models are available that specify $P(R|S_{ij})$, i.e., the probability of detection objects for a given sensor and location. Using Bayes theorem (4.10) it is now possible to update the grid model according to:

$$p_{ij}(t+1) = \frac{P(R|S_{ij} = O)p_{ij}(t)}{P(R|S_{ij} = O)p_{ij}(t) + P(R|S_{ij} = F)(1 - p_{ij}(t))},$$

where p_{ij} is computed across the grid model whenever new data are acquired.

The grid-based model has been widely used in mobile robotics [4.48, 49] and in medical imaging where image volumes are quite common [4.50]. Volume models can be relative large. As an example a millimeter-resolution grid model of the human head requires 4 GB of storage, and thus demands significant computational resources for maintenance.

4.4.3 Discrete Feature Representations

Both the raw sensor representation and the grid-based models contain a minimum of abstraction for the sensory data. In many cases there is an interest in extracting features from the sensor data to reduce the storage requirement and only preserve data that are invariant across motion of the platform or external objects. Features span most standard geometric entities such as points (\mathbf{p}), lines (\mathbf{l}), planes (\mathbf{N}, \mathbf{p}), curves ($\mathbf{p}(s)$), and more general surfaces. For estimation of properties of the external world there is a need for a hybrid model in which collections of features are integrated into a unified model of state.

In general a point is represented in $\mathcal{R}(3)$. Sensors have associated noise and, consequently, in most cases points have an associated uncertainty, typically modeled as Gaussian with mean μ and standard deviation σ . The estimation of the statistics is achieved using first- and second-order moments.

Line features are more difficult to represent. The mathematical line can be represented by the vector pair (\mathbf{p}, \mathbf{t}) , i.e., a point on the line and the tangent vector. In many practical applications the line has a finite extent, and there is a need to encode the length of the line,

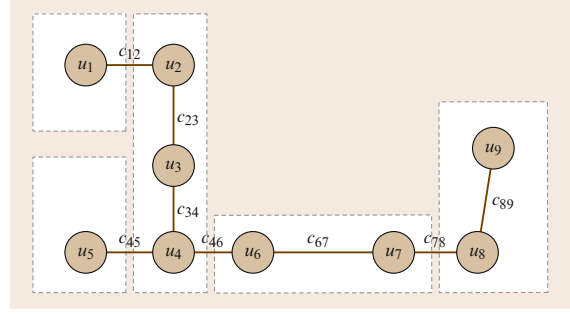


Fig. 4.16 A topological map of a spatial environment

which can be achieved using end points, start point, tangent, and length. In some cases it is advantageous to have a redundant representation of the line model to simplify updating and matching. The relation between end-point uncertainties and other line parameters can be derived analytically, as described in [4.51]. The estimation of line parameters is often based on the previously describe RANSAC method through the use of the Hough transform [4.8], which is another voting-based method.

For more complex feature models such as curves or surfaces there is a corresponding need to utilize detection methods that facilitate robust segmentation of features, and estimation of the associated uncertainty. A comprehensive description of such methods is available from [4.36].

4.4.4 Symbolic/Graph-Based Models

All of the representations presented in Sects. 4.4.1–4.4.3 are parametric in nature with limited associated semantics. Methods for the recognition of structures, spaces, locations, and objects have seen major recent progress in particular due to advances in statistical learning theory [4.10, 52]. Consequently, today there exist a variety of methods for the recognition of complex structures in sensor data, such as landmarks, road surfaces, body structures, etc. Given the availability of recognized structures it is possible to represent the environment using the previously discussed graphical models. In general a graph is composed of a set of nodes N and a set of edges E that connect nodes. Both nodes and edges can have attributes associated such as labels and distances. One example of a graph structure is a topological map of the environment as shown in Fig. 4.16. The graph representation could also be a semantic model of the environment (objects and places) or a representation of the composition of an object to assembled.

In terms of model updating semantic/graph-based representations can take advantage of recent advances

in Bayesian reasoning as presented by *Pearl* [4.53], and exemplified in [4.54].

4.5 Conclusions and Further Readings

Sensing and estimation continues to be a challenging and very active area of robotics research. Several areas of sensing such as computer vision and medical imaging are themselves large and diverse research areas. At the same time, new fundamental and applied techniques in estimation continue to be developed. Indeed, it is fair to say that perception continues to be one of the most challenging areas of robotics research.

Given this wealth of activity, no single chapter can hope to cover all of the material that can be useful in the development of sensor-based robotics. However, the methods that have been presented here are representative of the most commonly used techniques in robotics. In particular, linear techniques such as the Kalman filter continue to form the backbone of perceptive robotics. Part C of the handbook provides more in-depth coverage of several of the key topics in sensing and estimation.

For the reader wishing to learn more, general discussion on the design, physics, and use of a rich variety of sensors can be found in the Handbook of Modern Sensors [4.3]. A discussion of sensors for mobile robots can be found in [4.55], though significant advances have been achieved since the book was published more than a decade ago. Sensing and estimation using computer vision is described in detail in [4.56] and [4.57].

The basic estimation theory is covered in a number of excellent text books. Much of the detection and linear estimation theory is covered in depth in [4.18] and [4.58]. General statistical estimation is covered in [4.10] and [4.11] and the more recently updated version [4.36]. Robust methods are described in detail in [4.19, 35]. In-depth coverage of estimation methods for mobile systems is also covered in [4.31].

References

- 4.1 D.C. Marr: *Vision* (Freeman, Bedford 1982)
- 4.2 R. Siegwart, I.R. Nourbakhsh: Autonomous mobile robots. In: *Intelligent Robotics and Autonomous Systems* (MIT Press, Cambridge 2004)
- 4.3 J. Fraden: *Handbook of Modern Sensors: Physic, Design and Applications*, 2nd edn. (Springer, New York 1996)
- 4.4 G. Dissanayaka, S. Sukkarieh, E. Nebot, H. Durrant-Whyte: The aiding of a low-cost strapdown inertial measurement unit using vehicle model constraints for land vehicle applications, *IEEE Trans. Robot. Autom.* **17**(5), 731–748 (2001)
- 4.5 Z. Zhang: A flexible new technique for camera calibration, *IEEE Trans. Pattern. Anal. Mach. Intell.* **22**(11), 1330–1334 (2000)
- 4.6 D. Burschka, J. Geiman, G.D. Hager: Optimal landmark configuration for vision-based control of mobile robots, *Proc. Int. Conf. Robot. Autom. (ICRA 2003)* pp. 3917–3922
- 4.7 J. Baker, N. Anderson, P. Pillis: Ground-penetrating radar surveying in support of archeological site investigations, *Comput. Geosci.* **23**(10), 1093–1099 (1997)
- 4.8 P. V. C. Hough: A method and means for recognizing complex patterns, U.S. Patent 3,069,654 (1962)
- 4.9 M.A. Fischler, R.C. Bolles: Random Sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM* **24**, 381–395 (1981)
- 4.10 T. Hastie, R. Tibshirani, J. Friedman: *The Elements of Statistical Learning*, Springer Series in Statistics (Springer, Berlin, Heidelberg 2002)
- 4.11 R.O. Duda, P.E. Hart: *Pattern Classification and Scene Analysis* (Wiley-Interscience, New York 1973)
- 4.12 R. Vidal, Y. Ma, J. Piazzi: A new GPCA algorithm for clustering subspaces by fitting, differentiating and dividing polynomials, *Proc. Int. Conf. Comput. Vis. Pattern Recog.* **1**, 510–517 (2004)
- 4.13 P. Besl, N.D. McKay: A method for registration of 3-D shapes, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 239–256 (1992)
- 4.14 F. Dellaert, S. Seitz, C. Thorpe, S. Thrun: Special issue on Markov chain Monte Carlo methods, *Mach. Learn.* **50**, 45–71 (2003)
- 4.15 A. Gelb (Ed.): *Applied Optimal Estimation* (MIT Press, Cambridge 1974)
- 4.16 D. Simon: *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches* (Wiley, New York 2006)
- 4.17 A. Doucet, N. de Freitas, N. Gordon: *Sequential Monte Carlo Methods in Practice* (Springer, Berlin, Heidelberg 2001)
- 4.18 Y. Bar-Shalom, T. Fortmann: *Tracking and Data Association* (Academic, New York 1988)

- 4.19 P.J. Huber: *Robust Statistics* (Wiley, New York 1981)
- 4.20 J.O. Berger: *Statistical Decision Theory and Bayesian Analysis*, 2nd edn. (Springer, New York 1985)
- 4.21 G.D. Hager: *Task-Directed Sensor Fusion and Planning* (Kluwer, Boston 1990)
- 4.22 S. Abrams, P.K. Allen, K. Tarabanis: Computing camera viewpoints in a robot work-cell, *Int. J. Robot. Res.* **18**(3), 267–285 (1999)
- 4.23 M. Suppa, P. Wang, K. Gupta, G. Hirzinger: C-space exploration using noisy sensor models, *Proc. IEEE Int. Conf. Robot. Autom.* (2004) pp.1927–1932
- 4.24 G.S. Chirikjian, A.B. Kyatkin: *Engineering Applications of Noncommutative Harmonic Analysis* (CRC, Boca Raton 2000)
- 4.25 J.C. Kinsey, L.L. Whitcomb: Adaptive identification on the group of rigid body rotations and its application to precision underwater robot navigation, *IEEE Trans. Robot.* **23**, 124–136 (2007)
- 4.26 P.J. Bickel, K.A. Doksum: *Mathematical Statistics*, 2nd edn. (Prentice-Hall, Upper Saddle River 2006)
- 4.27 G. Strang: *Linear Algebra and its Applications*, 4th edn. (Brooks Cole, New York 2005)
- 4.28 P. McCullagh, J.A. Nelder: *Generalized Linear Models*, 2nd edn. (Chapman Hall, New York 1989)
- 4.29 E.L. Lehmann, G. Casella: *Theory of Point Estimation* (Springer, New York 1998)
- 4.30 R.E. Kalman: A new approach to linear filtering and prediction problems, *Transactions of the ASME, J. Basic Eng.* **82**, 35–45 (1960)
- 4.31 S. Thrun, D. Fox, W. Burgard: *Probabilistic Robotics, Autonomous Robotics and Intelligent Agents* (MIT Press, Cambridge 2005)
- 4.32 C. Bishop: *Pattern Recognition and Machine Learning* (Springer, New York 2006)
- 4.33 G.D. Hager, P.N. Belhumeur: Efficient region tracking of with parametric models of illumination and geometry, *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(10), 1025–1039 (1998)
- 4.34 J.W. Hardin, J.M. Hilbe: *Generalized Linear Models and Extensions*, 2nd edn. (Stata, College Station 2007)
- 4.35 P.J. Rousseeuw, A. Leroy: *Robust Regression and Outlier Detection* (Wiley, New York 1987)
- 4.36 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification*, 2nd edn. (Wiley, New York 2001)
- 4.37 P.C. Mahalanobis: On the generalised distance in statistics, *Proc. Nat. Inst. Sci. India* **12**, 49–55 (1936)
- 4.38 J. Hamilton: *Time Series Analysis* (Princeton Univ. Press, Princeton 1994)
- 4.39 S. Atiya, G.D. Hager: Real-time vision-based robot localization, *IEEE Trans. Robot. Autom.* **9**(6), 785–800 (1993)
- 4.40 G.D. Hager: Task-directed computation of qualitative decisions from sensor data, *IEEE Trans. Robot. Autom.* **10**(4), 415–429 (1994)
- 4.41 G. Shafer: *A Mathematical Theory of Evidence* (Princeton Univ. Press, Princeton 1976)
- 4.42 J. Rosenblatt: DAMN: A distributed architecture for mobile navigation, *AAAI 1995 Spring Symposium on Lessons Learned for Implementing Software Architectures for Physical Agents* (1995) pp.167–178
- 4.43 R.M. Murrey, Z. Li, S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1993)
- 4.44 K.J. Åström, B. Wittenmark: *Adaptive Control*, 2nd edn. (Addison-Wesley, Reading 1995)
- 4.45 S. Gutmann, C. Schlegel: AMOS: Comparison of scan-matching approaches for self-localization in indoor environments, *1st Euromicro Conf. Adv. Mobile Robotics* (1996)
- 4.46 S. Gutmann: *Robust Navigation for Autonomous Mobile Systems*. Ph.D. Thesis (Alfred Ludwig University, Freiburg 2000)
- 4.47 H. Samet: The quadtree and related hierarchical data structures, *ACM Comput. Surv.* **16**(2), 187–260 (1984)
- 4.48 A. Elfes: Sonar-based real-world mapping and navigation, *IEEE Trans. Robot. Autom.* **3**(3), 249–265 (1987)
- 4.49 A. Elfes: *A Probabilistic Framework for Robot Perception and Navigation*. Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1989)
- 4.50 M.R. Stytz, G. Frieder, O. Frieder: Three-dimensional medical imaging: Algorithms and computer systems, *ACM Comput. Surv.* **23**(4), 421–499 (1991)
- 4.51 R. Deriche, R. Vaillant, O. Faugeras: From Noisy Edges Points to 3D Reconstruction of a Scene: A robust approach and its uncertainty analysis. In: *Theory and Applications of Image Analysis* (World Scientific Singapore 1992) pp. 71–79
- 4.52 V.N. Vapnik: *Statistical Learning Theory* (Wiley, New York 1998)
- 4.53 J. Pearl: *Probabilistic Reasoning in Intelligent Systems* (Morgan Kaufmann, New York 1988)
- 4.54 M. Paskin: *Thin Junction Tree Filters for Simultaneous Localisation and Mapping*. Ph.D. Thesis (University of California, Berkley 2002)
- 4.55 H.R. Everett: *Sensors for Mobile Robots: Theory and Application* (Peters, London 1995)
- 4.56 D. Forsyth, J. Ponce: *Computer Vision – A Modern Approach* (Prentice-Hall, Upper Saddle River 2003)
- 4.57 R. Hartley, A. Zisserman: *Multiple View Geometry in Computer Vision* (Cambridge Univ. Press, Cambridge 2000)
- 4.58 S. Blackman, R. Popoli: *Design and Analysis of Modern Tracking Systems* (Artech House, London 1999)

Motion Planning

5. Motion Planning

Lydia E. Kavraki, Steven M. LaValle

This chapter first provides a formulation of the geometric path planning problem in Sect. 5.1 and then introduces sampling-based planning in Sect. 5.2. Sampling-based planners are general techniques applicable to a wide set of problems and have been successful in dealing with hard planning instances. For specific, often simpler, planning instances, alternative approaches exist and are presented in Sect. 5.3. These approaches provide theoretical guarantees and for simple planning instances they outperform sampling-based planners. Section 5.4 considers problems that involve differential constraints, while Sect. 5.5 overviews several other extensions of the basic problem formulation and proposed solutions. Finally, Sect. 5.7 addresses some important and more advanced topics related to motion planning.

5.1 Motion Planning Concepts	110
5.1.1 Configuration Space	110
5.1.2 Geometric Path Planning Problem ..	111
5.1.3 Complexity of Motion Planning	111
5.2 Sampling-Based Planning	111
5.2.1 Multi-Query Planners: Mapping the Connectivity of $\mathcal{C}_{\text{free}}$..	112

5.2.2 Single-Query Planners: Incremental Search	113
5.3 Alternative Approaches	115
5.3.1 Combinatorial Roadmaps	115
5.3.2 Roadmaps in Higher Dimensions	116
5.3.3 Potential Fields	117
5.4 Differential Constraints	118
5.4.1 Concepts and Terminology	118
5.4.2 Discretization of Constraints	119
5.4.3 Decoupled Approach	119
5.4.4 Kinodynamic Planning	120
5.5 Extensions and Variations	121
5.5.1 Closed Kinematic Chains	121
5.5.2 Manipulation Planning	122
5.5.3 Time-Varying Problems	122
5.5.4 Multiple Robots	122
5.5.5 Uncertainty in Predictability	123
5.5.6 Sensing Uncertainty	124
5.6 Advanced Issues	124
5.6.1 Topology of Configuration Spaces ...	124
5.6.2 Sampling Theory	125
5.6.3 Computational Algebraic Geometry Techniques	126
5.7 Conclusions and Further Reading	127
References	128

A fundamental robotics task is to plan collision-free motions for complex bodies from a start to a goal position among a collection of static obstacles. Although relatively simple, this geometric path planning problem is computationally hard [5.1]. Extensions of this formulation take into account additional problems that are inherited from mechanical and sensor limitations of real robots such as uncertainties, feedback, and differential constraints, which further complicate the development of automated planners. Modern al-

gorithms have been fairly successful in addressing hard instances of the basic geometric problem and a lot of effort has been devoted to extend their capabilities to more challenging instances. These algorithms have had widespread success in applications beyond robotics, such as computer animation, virtual prototyping, and computational biology. There are many available surveys [5.2–4] and books [5.5–7] that cover modern motion planning techniques and applications.

5.1 Motion Planning Concepts

This section provides a description of the fundamental motion planning problem or the geometric path planning problem. Extensions of this basic formulation to more complicated instances will be discussed later in the chapter and will be revisited throughout this book.

5.1.1 Configuration Space

In path planning, a complete description of the geometry of a robot \mathcal{A} and of a workspace \mathcal{W} is provided. The workspace $\mathcal{W} = \mathbb{R}^N$, in which $N = 2$ or $N = 3$, is a static environment populated with obstacles. The goal is to find a collision-free path for \mathcal{A} to move from an initial position and orientation to a goal position and orientation.

To achieve this, a complete specification of the location of every point on the robot geometry, or a *configuration* \mathbf{q} , must be provided. The *configuration space*, or C-space ($\mathbf{q} \in \mathcal{C}$), is the space of all possible configurations. The C-space represents the set of all transformations that can be applied to a robot given its kinematics as described in Chap. 1 (Kinematics). It was recognized early on in motion planning research [5.8, 9] that the C-space is a useful way to abstract planning problems in a unified way. The advantage of this abstraction is that a robot with a complex geometric shape is mapped to a single point in the C-space. The number of degrees of freedom of a robot system is the dimension of the C-space, or the minimum number of parameters needed to specify a configuration.

Let the closed set $\mathcal{O} \subset \mathcal{W}$ represent the (*workspace*) *obstacle region*, which is usually expressed as a collection of polyhedra, three-dimensional (3-D) triangles, or piecewise-algebraic surfaces. Let the closed set $\mathcal{A}(\mathbf{q}) \subset \mathcal{W}$ denote the set of points occupied by the robot when at configuration $\mathbf{q} \in \mathcal{C}$; this set is usually modeled using the same primitives as used for \mathcal{O} . The *C-space obstacle region*, \mathcal{C}_{obs} , is defined as

$$\mathcal{C}_{\text{obs}} = \{\mathbf{q} \in \mathcal{C} \mid \mathcal{A}(\mathbf{q}) \cap \mathcal{O} \neq \emptyset\}. \quad (5.1)$$

Since \mathcal{O} and $\mathcal{A}(\mathbf{q})$ are closed sets in \mathcal{W} , the obstacle region is a closed set in \mathcal{C} . The set of configurations that avoid collision is $\mathcal{C}_{\text{free}} = \mathcal{C} \setminus \mathcal{C}_{\text{obs}}$, and is called the *free space*.

Simple Examples of C-spaces

Translating Planar Rigid Bodies. The robot's configuration can be specified by a reference point (x, y) on the planar rigid body relative to some fixed coordinate

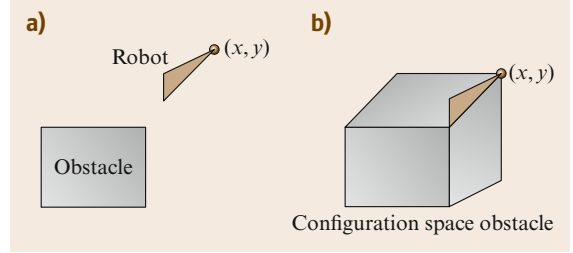


Fig. 5.1a,b A robot translating in the plane: (a) a triangular robot moves in a workspace with a single rectangular obstacle. (b) The C-space obstacle

frame. Therefore the C-space is equivalent to \mathbb{R}^2 . Figure 5.1 gives an example of a C-space for a triangular robot and a single polygonal obstacle. The obstacle region in the C-space can be traced by sliding the robot around the workspace obstacle to find the constraints on all $\mathbf{q} \in \mathcal{C}$. Motion planning for the robot is now equivalent to motion planning for a point in the C-space.

Planar Arms. Figure 5.2 gives an example of a two-joint planar arm. The bases of both links are pinned, so that they can only rotate around the joints, and there are no joint limits. For this arm, specifying the rotational parameters θ_1 and θ_2 provides the configuration. Each joint angle θ_i corresponds to a point on the unit circle \mathbb{S}^1 and the C-space is $\mathbb{S}^1 \times \mathbb{S}^1 = T^2$, the two-dimensional torus shown in Fig. 5.2. For a higher number of links without joint limits, the C-space can be similarly defined as:

$$\mathcal{C} = \mathbb{S}^1 \times \mathbb{S}^1 \times \cdots \times \mathbb{S}^1. \quad (5.2)$$

If a joint has limits, then each corresponding \mathbb{S}^1 is often replaced with \mathbb{R} , even though it is a finite interval. If the base of the planar arm is mobile and not pinned,

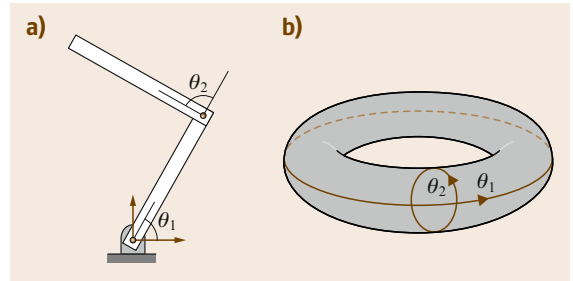


Fig. 5.2 (a) A two-joint planar arm in which the links are pinned and there are no joint limits. (b) The C-space

then the additional translation parameters must also be considered in the arm's configuration:

$$\mathcal{C} = \mathbb{R}^2 \times \mathbb{S}^1 \times \mathbb{S}^1 \times \dots \times \mathbb{S}^1. \quad (5.3)$$

Additional examples of C-spaces are provided in Sect. 5.6.1, where topological properties of configuration spaces are discussed.

5.1.2 Geometric Path Planning Problem

The basic motion planning problem, also known as the *piano mover's problem* [5.1], is defined as follows. Given:

1. A *workspace* \mathcal{W} , where either $\mathcal{W} = \mathbb{R}^2$ or $\mathcal{W} = \mathbb{R}^3$.
2. An *obstacle region* $\mathcal{O} \subset \mathcal{W}$.
3. A *robot* defined in \mathcal{W} . Either a rigid body \mathcal{A} or a collection of m links: $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_m$.
4. The *configuration space* \mathcal{C} (\mathcal{C}_{obs} and $\mathcal{C}_{\text{free}}$ are then defined).
5. An *initial configuration* $\mathbf{q}_I \in \mathcal{C}_{\text{free}}$.
6. A *goal configuration* $\mathbf{q}_G \in \mathcal{C}_{\text{free}}$. The initial and goal configuration are often called a *query* ($\mathbf{q}_I, \mathbf{q}_G$).

Compute a (continuous) path, $\tau : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$, such that $\tau(0) = \mathbf{q}_I$ and $\tau(1) = \mathbf{q}_G$.

5.1.3 Complexity of Motion Planning

The main complications in motion planning are that it is not easy to directly compute \mathcal{C}_{obs} and $\mathcal{C}_{\text{free}}$, and the dimensionality of the C-space is often quite high. In terms of computational complexity, the piano mover's problem was studied early on and it was shown to be PSPACE-hard by Reif [5.1]. A series of polynomial-time algorithms for problems with fixed dimension suggested an exponential dependence on the problem dimensionality [5.10, 11]. A single exponential-time algorithm in the C-space dimensionality was proposed by Canny and

showed that the problem is PSPACE-complete [5.12]. Although impractical, the algorithm serves as an upper bound on the general version of the basic motion planning problem. It applies computational algebraic geometry techniques for modeling the C-space in order to construct a *roadmap*, a one-dimensional (1-D) subspace that captures the connectivity of $\mathcal{C}_{\text{free}}$. Additional details about such techniques can be found in Sect. 5.6.3.

The complexity of the problem motivated work in path planning research. One direction was to study subclasses of the general problem for which polynomial time algorithms exist [5.13]. Even some simpler, special cases of motion planning, however, are at least as challenging, for example, the case of a finite number of translating, axis-aligned rectangles in \mathbb{R}^2 is PSPACE-hard as well [5.14]. Some extensions of motion planning are even harder. For example, a certain form of planning under uncertainty in 3-D polyhedral environment is NEXPTIME-hard [5.15]. The hardest problems in NEXPTIME are believed to require doubly exponential time to solve.

A different direction was the development of alternative planning paradigms that were practical under realistic assumptions. Many combinatorial approaches can efficiently construct 1-D roadmaps for specific two-dimensional (2-D) or 3-D problems. Potential field-based approaches define vector fields which can be followed by a robot towards the goal. Both approaches, however, do not scale well in the general case. They will be described in Sect. 5.3. An alternative paradigm, sampling-based planning, is a general approach that has been shown to be successful in practice for many challenging problems. It avoids the exact geometric modeling of the C-space but it cannot provide the guarantees of a complete algorithm. Complete and exact algorithms are able to detect that no path can be found. Instead sampling-based planning offers a lower level of completeness guarantee. This paradigm is described in the following section.

5.2 Sampling-Based Planning

Sampling-based planners are described first because they are the method of choice for a very general class of problems. The following section will describe other planners, some of which were developed before the sampling-based framework. The key idea in sampling-based planning is to exploit advances in collision detection algorithms that compute whether

a single configuration is collision free. Given this simple primitive, a planner samples different configurations to construct a data structure that stores 1-D C-space curves, which represent collision-free paths. In this way, sampling-based planners do not access the C-space obstacles directly but only through the collision detector and the constructed data struc-

ture. Using this level of abstraction, the planners are applicable to a wide range of problems by tailoring the collision detector to specific robots and applications.

A standard for sampling-based planners is to provide a weaker, but still interesting, form of completeness: *if a solution path exists, the planner will eventually find it*. Giving up on the stronger form of completeness, which also requires failure to be reported in finite time, these techniques are able to solve in practice problems with more than three degrees of freedom that complete approaches cannot address. More details on this weaker form of completeness are provided in Sect. 5.6.2.

Different planners follow different approaches on how to sample configurations and what kind of data structures they construct. Section 5.6.2 provides a deeper insight on sampling issues. A typical classification of sampling-based planners is between multi-query and single-query approaches:

- In the first category, the planners construct a roadmap, an undirected graph G that is precomputed once so as to *map the connectivity properties of $\mathcal{C}_{\text{free}}$* . After this step, multiple queries in the same environment can be answered using only the constructed roadmap. Such planners are described in Sect. 5.2.1.
- Planners in the second category build tree data structures on the fly given a planning query. They attempt to focus on exploring the part of the C-space that will lead to solving a specific query as fast as possible. They are described in Sect. 5.2.2.

Both approaches, however, make similar use of a collision checking primitive. The objective of a collision detector is to report all geometric contacts between objects given their geometries and transformations [5.16–18]. The availability of packages that were able to answer collision queries in a fraction of a second was critical to the development of sampling-based planners. Modern planners use collision detectors as a *black box*. Initially the planner provides the geometries of all the involved objects and specifies which of them are mobile. Then, in order to validate a robot configuration, a planner provides the corresponding robot transformation and a collision detector responds on whether the objects collide or not. Many packages represent the geometric models hierarchically, avoid computing all-pairwise interactions, and conduct a binary search to evaluate collisions. Except from configurations, a planner must also validate

entire paths. Some collision detectors return distance-from-collision information, which can be used to infer that entire neighborhoods in \mathcal{C} are valid. It is often more expensive, however, to extract this information; instead paths are usually validated point by point using a small stepping size either incrementally or by employing binary search. Some collision detectors are incremental by design, which means that they can be faster by reusing information from a previous query [5.16].

5.2.1 Multi-Query Planners: Mapping the Connectivity of $\mathcal{C}_{\text{free}}$

Planners that aim to answer multiple queries for a certain static environment use a preprocessing phase during which they attempt to map the connectivity properties of $\mathcal{C}_{\text{free}}$ onto a roadmap. This roadmap has the form of a graph G , with vertices as configurations and edges as paths. A union of 1-D curves is a roadmap G if it satisfies the following properties:

1. *Accessibility*: From any $q \in \mathcal{C}_{\text{free}}$, it is simple and efficient to compute a path $\tau : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$ such that $\tau(0) = q$ and $\tau(1) = s$, in which s may be any point in $S(G)$. $S(G)$ is the *swath* of G , the union of all configurations reached by all edges and vertices. This means that it is always possible to connect a planning query pair q_I and q_G to some s_I and s_G , respectively, in $S(G)$.
2. *Connectivity preserving*: The second condition requires that, if there exists a path $\tau : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$ such that $\tau(0) = q_G$ and $\tau(1) = q_I$, then there also exists a path $\tau' : [0, 1] \rightarrow S(G)$, such that $\tau'(0) = s_I$ and $\tau'(1) = s_G$. Thus, solutions are not missed because G fails to capture the connectivity of $\mathcal{C}_{\text{free}}$.

The probabilistic roadmap method (PRM) approach [5.19] attempts to approximate such a roadmap G in a computationally efficient way. The preprocessing phase of PRM, which can be extended to sampling-based roadmaps in general, follows these steps:

1. *Initialization*: Let $G(V, E)$ represent an undirected graph, which is initially empty. Vertices of G will correspond to collision-free configurations, and edges to collision-free paths that connect vertices.
2. *Configuration sampling*: A configuration $\alpha(i)$ is sampled from $\mathcal{C}_{\text{free}}$ and added to the vertex set V . $\alpha(\cdot)$ is an infinite, dense sample sequence and $\alpha(i)$ is the i -th point in that sequence.

3. *Neighborhood computation*: Usually, a *metric* is defined in the C-space, $\rho : \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$. Vertices q already in V are then selected as part of $\alpha(i)$'s neighborhood if they have small distance according to ρ .
4. *Edge consideration*: For those vertices q that do not belong in the same connected component of G with $\alpha(i)$ the algorithm attempts to connect them with an edge.
5. *Local planning method*: Given $\alpha(i)$ and $q \in \mathcal{C}_{\text{free}}$ a module is used that attempts to construct a path $\tau_s : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$ such that $\tau(0) = \alpha(i)$ and $\tau(1) = q$. Using collision detection, τ_s must be checked to ensure that it does not cause a collision.
6. *Edge insertion*: Insert τ_s into E , as an edge from $\alpha(i)$ to q .
7. *Termination*: The algorithm is typically terminated when a predefined number of collision-free vertices N has been added in the roadmap.

The algorithm is incremental in nature. Computation can be repeated by starting from an already existing graph. A general sampling-based roadmap is summarized in Algorithm 1.

Algorithm 5.1

SAMPLING-BASED ROADMAP

N : number of nodes to include in the roadmap

```

G.init(); i ← 0;
while i < N do
  if  $\alpha(i) \in \mathcal{C}_{\text{free}}$  then
    G.add_vertex( $\alpha(i)$ ); i ← i + 1;
    for  $q \in \text{NEIGHBORHOOD}(\alpha(i), G)$  do
      if CONNECT( $\alpha(i), q$ ) then
        G.add_edge( $\alpha(i), q$ );
      endif
    endfor
  endif
endwhile

```

An illustration of the algorithm's behavior is depicted in Fig. 5.3. To solve a query, q_I and q_G are connected to the roadmap, and graph search is performed.

For the original PRM [5.19], the configuration $\alpha(i)$ was produced using random sampling. For the connection step between q and $\alpha(i)$, the algorithm used straight line paths in the C-space. In some cases a connection was not attempted if q and $\alpha(i)$ were in the same connected component. There have been many subsequent works that try to improve the roadmap

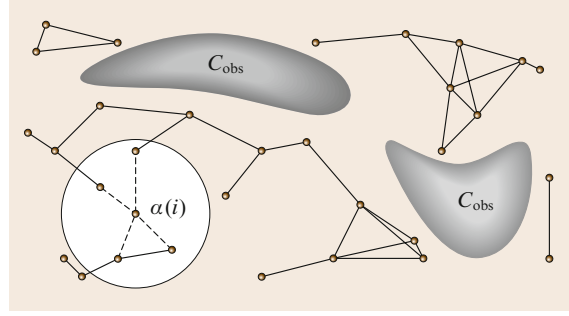


Fig. 5.3 The sampling-based roadmap is constructed incrementally by attempting to connect each new sample, $\alpha(i)$, to nearby vertices in the roadmap

quality while using fewer samples. Methods for concentrating samples at or near the boundary of $\mathcal{C}_{\text{free}}$ are presented in [5.20, 21]. Methods that move samples as far from the boundary as possible appear in [5.22, 23]. Deterministic sampling techniques, including grids, appear in [5.24]. A method of pruning vertices based on mutual visibility that leads to a dramatic reduction in the number of roadmap vertices appears in [5.25]. Theoretical analysis of sampling-based roadmaps appears in [5.24, 26, 27] and is briefly discussed in Sect. 5.6.2. An experimental comparison of sampling-based roadmap variants appears in [5.28]. One difficulty in these roadmap approaches is identifying narrow passages. One proposal is to use the *bridge test* for identifying these [5.29]. For other PRM-based works, see [5.30–34]. Extended discussion of the topic can be found in [5.5, 7].

5.2.2 Single-Query Planners: Incremental Search

Single-query planning methods focus on a single initial-goal configuration pair. They probe and search the continuous C-space by extending tree data structures initialized at these known configurations and eventually connecting them. Most single-query methods conform to the following template:

1. *Initialization*: Let $G(V, E)$ represent an undirected search graph, for which the vertex set V contains a vertex for one (usually q_I) or more configurations in $\mathcal{C}_{\text{free}}$, and the edge set E is empty. Vertices of G are collision-free configurations, and edges are collision-free paths that connect vertices.
2. *Vertex selection method*: Choose a vertex $q_{\text{cur}} \in V$ for expansion.

3. *Local planning method*: For some $\mathbf{q}_{\text{new}} \in \mathcal{C}_{\text{free}}$, which may correspond to an existing vertex in V but on a different tree or a sampled configuration, attempt to construct a path $\tau_s : [0, 1] \rightarrow \mathcal{C}_{\text{free}}$ such that $\tau(0) = \mathbf{q}_{\text{cur}}$ and $\tau(1) = \mathbf{q}_{\text{new}}$. Using collision detection, τ_s must be checked to ensure that it does not cause a collision. If this step fails to produce a collision-free path segment, then go to Step 2.
4. *Insert an edge in the graph*: Insert τ_s into E , as an edge from \mathbf{q}_{cur} to \mathbf{q}_{new} . If \mathbf{q}_{new} is not already in V , then it is inserted.
5. *Check for a solution*: Determine whether G encodes a solution path.
6. *Return to Step 2*: Iterate unless a solution has been found or some termination condition is satisfied, in which case the algorithm reports failure.

During execution, G may be organized into one or more trees. This leads to: (1) *unidirectional* methods, which involve a single tree, usually rooted at \mathbf{q}_I [5.35], (2) *bidirectional* methods, which involve two trees, typically rooted at \mathbf{q}_I and \mathbf{q}_G [5.35], and (3) *multidirectional* methods, which may have more than two trees [5.36,37]. The motivation for using more than one tree is that a single tree may become trapped trying to find an exit through a narrow opening. Traveling in the opposite direction, however, may be easier. As more trees are considered it becomes more complicated to determine which connections should be made between trees.

Rapidly Exploring Dense Trees

The important idea with this family of techniques is that the algorithm must incrementally explore the properties of the C-space. An algorithm that achieves this objective is the rapidly exploring random tree (RRT) [5.35], which can be generalized to the rapidly exploring dense tree (RDT) for any dense sampling, deterministic or random [5.7]. The basic idea is to induce a *Voronoi bias* in the exploration process by selecting for expansion the point in the tree that is closest to $\alpha(i)$ in each iteration. Using random samples, the probability that a vertex is chosen is proportional to the volume of its Voronoi region. The tree construction is outlined as:

Algorithm 5.2

RAPIDLY EXPLORING DENSE TREES

k : the exploration steps of the algorithm

```

G.init( $\mathbf{q}_I$ );
for  $i = 1$  to  $k$  do
    G.add_vertex( $\alpha(i)$ );
     $\mathbf{q}_n \leftarrow \text{NEAREST}(S(G), \alpha(i))$ ;
    G.add_edge( $\mathbf{q}_n, \alpha(i)$ );
end for

```

The tree starts at \mathbf{q}_I , and in each iteration, an edge and vertex are added.

So far, the problem of reaching \mathbf{q}_G has not been explained. There are several ways to use RDTs in a planning algorithm. One approach is to bias $\alpha(i)$ so that \mathbf{q}_G is frequently chosen (perhaps once every 50 iterations). A more efficient approach is to develop a bidirectional search by growing two trees, one from each of \mathbf{q}_I and \mathbf{q}_G . Roughly half of the time is spent expanding each tree in the usual way, while the other half is spent attempting to connect the trees. The simplest way to connect trees is to let the newest vertex of one tree be a substitute for $\alpha(i)$ in extending the other. This tricks one RDT into attempting to connect to the other while using the basic expansion algorithm [5.38]. Several works have extended, adapted, or applied RDTs in various applications [5.37, 39–42]. Detailed descriptions can be found in [5.5, 7].

Other Tree Algorithms

Planners based on the idea of expansive spaces are presented in [5.43–45]. In this case, the algorithm forces exploration by choosing vertices for expansion that have fewer points in a neighborhood around them. In [5.46], additional performance is obtained by self-tuning random walks, which focus virtually all of their effort on exploration. Other successful tree-based algorithms include the path-directed subdivision tree algorithm [5.47] and some of its variants [5.48]. In the literature, it is sometimes hard to locate tree-based planners for ordinary path planning problems as many of them (including RRT) were designed and/or applied to more complex problems (see Sect. 5.4.4). Their performance is nevertheless excellent for a variety of path planning problems.

5.3 Alternative Approaches

Alternative approaches to the sampling-based paradigm include potential-field-based techniques and combinatorial methods that also produce roadmaps, such as cell decompositions. These algorithms are able to elegantly and efficiently solve a narrow class of problems, and are much preferred over the algorithms of Sect. 5.2 in these cases. Most of the combinatorial algorithms are of theoretical interest, whereas sampling-based algorithms are motivated primarily by performance issues in challenging applications. Nevertheless, given some abstractions, the combinatorial algorithms can be used to solve practical problems such as autonomous navigation of mobile planar robots.

5.3.1 Combinatorial Roadmaps

Several algorithms exist for the case in which $\mathcal{C} = \mathbb{R}^2$ and \mathcal{C}_{obs} is polygonal. Most of these cannot be directly extended to higher dimensions; however, some of the general principles remain the same. The *maximum clearance roadmap* (or *retraction method* [5.49]) constructs a roadmap that keeps paths as far from the obstacles as possible. Paths are contributed to the roadmap from the three cases shown in Fig. 5.5, which correspond to all ways to pair together polygon features. The roadmap can be made naively in time $O(n^4)$ by generating all curves

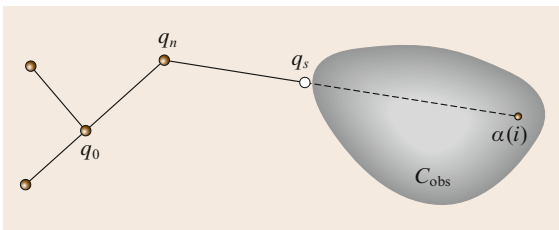


Fig. 5.4 If there is an obstacle, the edge travels up to the obstacle boundary, as far as allowed by the collision detection algorithm

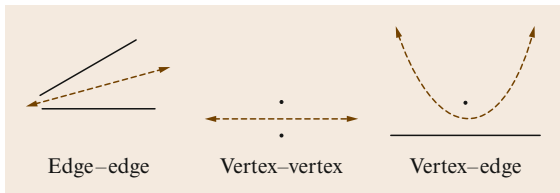


Fig. 5.5 Voronoi roadmap pieces are generated in one of three possible cases. The third case leads to a quadratic curve

shown in Fig. 5.5 for all possible pairs, computing their intersections, and tracing out the roadmap. Several algorithms exist that provide better asymptotic running time [5.50], but they are considerably more difficult to implement. The best-known algorithm runs in $O(n \lg n)$ time in which n is the number of roadmap curves [5.51].

An alternative is to compute a *shortest-path roadmap* [5.52], as shown in Fig. 5.6. This is different than the roadmap presented in the previous section because paths may actually touch the obstacles, which must be allowed for paths to be optimal. The roadmap vertices are the reflex vertices of \mathcal{C}_{obs} , which are vertices for which the interior angle is greater than π . An edge exists in the roadmap if and only if a pair of vertices is mutually visible and the line through them pokes into $\mathcal{C}_{\text{free}}$ when extended outward from each vertex (such lines are called *bitangents*). An $O(n^2 \lg n)$ -time construction algorithm can be formed by using a radial sweep algorithm from each reflex vertex. It can theoretically be computed in time $O(n^2 + m)$, in which m is the total number of edges in the roadmap [5.53].

Figure 5.7 illustrates the *vertical cell decomposition* approach. The idea is to decompose $\mathcal{C}_{\text{free}}$ into cells that are trapezoids or triangles. Planning in each cell is trivial because it is convex. A roadmap is made by placing a point in the center of each cell and each boundary between cells. Any graph search algorithm can be used to find a collision-free path quickly. The cell decomposition can be constructed in $O(n \lg n)$ time using the *plane-sweep principle* [5.54, 55]. Imagine that a vertical line sweeps from $x = -\infty$ to $x = \infty$, stopping at places where a polygon vertex is encountered. In these cases,

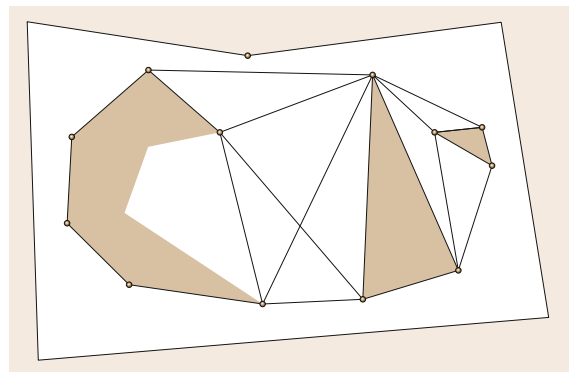


Fig. 5.6 The shortest-path roadmap includes edges between consecutive reflex vertices on \mathcal{C}_{obs} and also bitangent edges

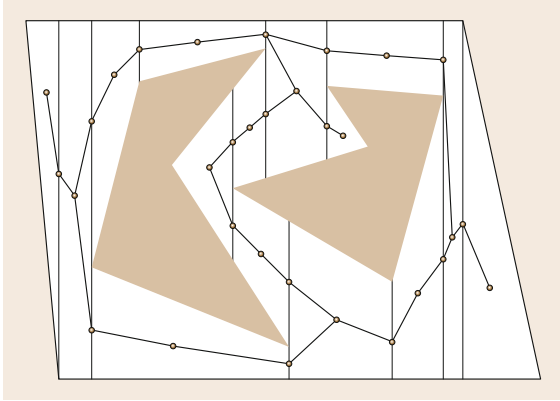


Fig. 5.7 The roadmap derived from the vertical cell decomposition

a cell boundary may be necessary above and/or below the vertex. The order in which segments stab the vertical line is maintained in a balanced search tree. This enables the determination of the vertical cell boundary limits in time $O(\lg n)$. The whole algorithm runs in time $O(n \lg n)$ because there are $O(n)$ vertices at which the sweep line can stop (also, the vertices need to be sorted at the outset, which requires time $O(n \lg n)$).

5.3.2 Roadmaps in Higher Dimensions

It would be convenient if the methods of Sect. 5.3.1 directly extend into higher dimensions. Although this unfortunately does not occur, some of the general ideas extend. To consider a cell decomposition in higher dimensions, there are two main requirements: (1) each cell should be simple enough that motion planning within a cell is trivial; (2) the cells should fit together nicely. A sufficient condition for the first requirement is that cells are convex; more general shapes may be allowed; however, the cells should not contain holes under any circumstances. For the second requirement, a sufficient condition is that the cells can be organized into a *singular complex*. This means that for any two d -dimensional cells for $d \leq n$, if the boundaries of the cells intersect, then the common boundary must itself be a complete cell (of lower dimension).

In two-dimensional polygonal C-spaces, triangulation methods define nice cell decompositions that are appropriate for motion planning. Finding good triangulations, which for example means trying to avoid thin triangles, is given considerable attention in computational geometry [5.55]. Determining the decomposition of a polygonal obstacle region with

holes that uses the smallest number of convex cells is *NP-hard* [5.56]. Therefore, we are willing to tolerate nonoptimal decompositions.

In three-dimensional C-spaces, if \mathcal{C}_{obs} is polyhedral, then the vertical decomposition method directly extends

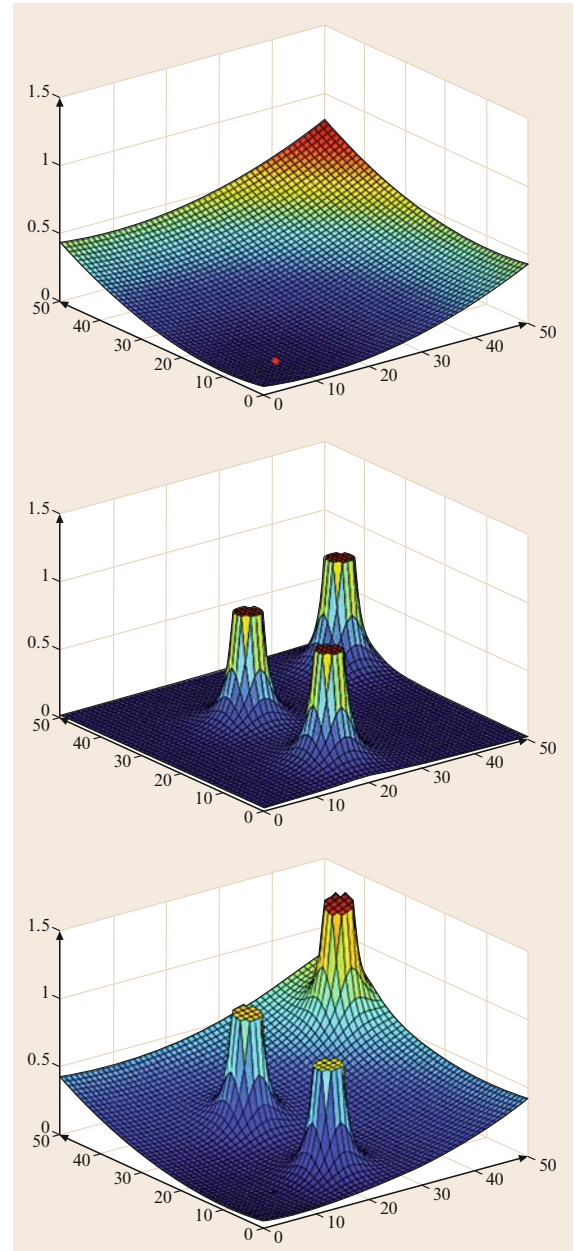


Fig. 5.8 An attractive and a repulsive component define a potential function

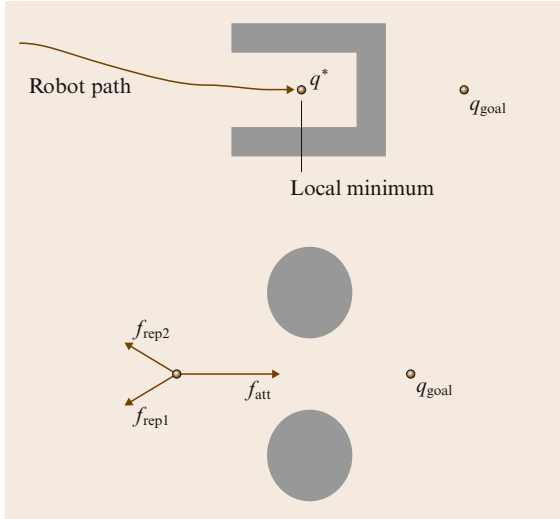


Fig. 5.9 Two examples of the local minimum problem with potential functions

by applying the plane sweep recursively, for example, the critical events may occur at each z coordinate, at which point changes a 2-D vertical decomposition over the x and y coordinates are maintained. The polyhedral case is obtained for a translating polyhedral robot among polyhedral obstacles in \mathbb{R}^3 ; however, for most interesting problems, \mathcal{C}_{obs} becomes nonlinear. Suppose $\mathcal{C} = \mathbb{R}^2 \times \mathbb{S}^1$, which corresponds to a robot that can translate and rotate in the plane. Suppose the robot and obstacles are polygonal. For the case of a line-segment robot, an $O(n^5)$ algorithm that is not too difficult to implement is given in [5.57]. The approaches for more general models and C-spaces are extremely difficult to use in practice; they are mainly of theoretical interest and are summarized in Sect. 5.6.3.

5.3.3 Potential Fields

A different approach for motion planning is inspired from obstacle avoidance techniques [5.58]. It does not explicitly construct a roadmap, but instead constructs a differentiable real-valued function $U : \mathbb{R}^m \rightarrow \mathbb{R}$, called a potential function, that guides the motion of the moving object. The potential is typically constructed so that it consists of an attractive component $U_a(\mathbf{q})$, which pulls the robot towards the goal, and a repulsive component $U_r(\mathbf{q})$, which pushes the robot away from the obstacles, as shown in Fig. 5.8. The gradient of the potential function is the vector $\nabla U(\mathbf{q}) = DU(\mathbf{q})^\top = [\frac{\partial U}{\partial q_1}(\mathbf{q}), \dots, \frac{\partial U}{\partial q_m}(\mathbf{q})]^\top$, which

points in the direction that locally maximally increases U . After the definition of U , a path can be computed by starting from \mathbf{q}_1 and applying *gradient descent*:

1. $\mathbf{q}(0) = \mathbf{q}_1; i = 0;$
2. **while** $\nabla U(\mathbf{q}(i)) \neq 0$ **do**
3. $\mathbf{q}(i+1) = \mathbf{q}(i) + \nabla U(\mathbf{q}(i))$
4. $i = i + 1$

However, this gradient-descent approach does not guarantee a solution to the problem. Gradient descent can only reach a local minimum of $U(\mathbf{q})$, which may not correspond to the goal state \mathbf{q}_G , as shown in Fig. 5.9.

A planner that makes use of potential functions and attempts to avoid the issue of local minima is the *randomized potential planner* [5.59]. The idea is to combine potential functions with random walks by employing multiple planning modes. In one mode, gradient descent is applied until a local minimum is reached. Another mode uses random walks to try to escape local minima. A third mode performs backtracking whenever several attempts to escape a local minimum have failed. In many ways, this approach can be considered as a sampling-based planner. It also provides the weaker

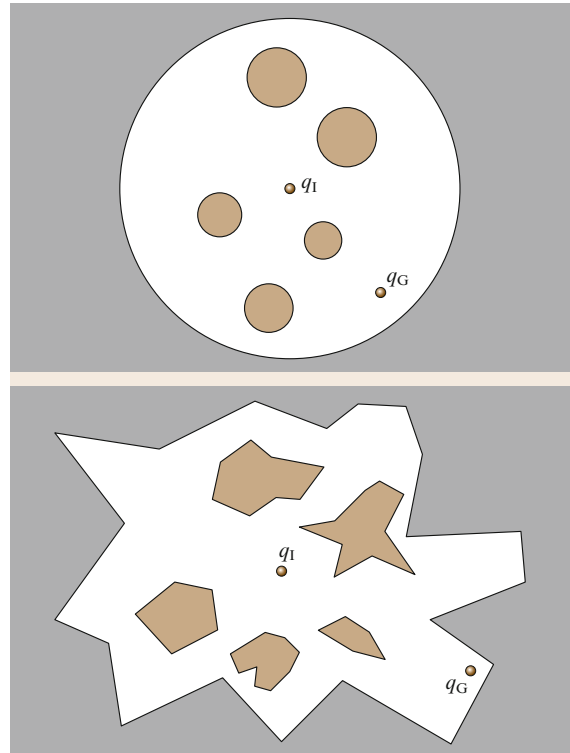


Fig. 5.10 Examples of sphere and star spaces

completeness guarantee but it requires parameter tuning. Recent sampling-based methods achieve better performance by spending more time exploring the space, rather than focusing heavily on a potential function.

The gradient of the potential function can be also used to define a *vector field*, which assigns a motion for the robot at any arbitrary configuration $\mathbf{q} \in \mathcal{C}$. This is an important advantage of the approach, beyond its computational efficiency, since it does not only compute a single path, but also a *feedback control* strategy. This makes the approach more robust against control and sensing errors. Most of the techniques in feedback motion planning are based on the idea of *navigation functions* [5.60], which are potential functions properly constructed so as to have a single minimum. A function $\phi : \mathcal{C}_{\text{free}} \rightarrow [0, 1]$ is called a navigation function if it:

- is smooth (or at least C^k for $k \geq 2$),
- has a unique minimum at \mathbf{q}_G in the connected component of the free space that contains \mathbf{q}_G ,

- is uniformly maximal on the free-space boundaries,
- and is Morse, which means that all its critical points, such as saddle points, are isolated and can be avoided with small random perturbations.

Navigation functions can be constructed for sphere boundary spaces centered at \mathbf{q}_1 that contain only spherical obstacles, as illustrated in Fig. 5.10. Then they can be extended to a large family of C-spaces that are diffeomorphic to sphere spaces, such as star-shaped spaces, as shown in Fig. 5.10. A more elaborate description of strategies for feedback motion planning will be presented in Chaps. 35, 36, and 37.

Putting the issue of local minima aside, another major challenge for such potential function based approaches is constructing and representing the C-space in the first place. This issue makes the applications of these techniques too complicated for high-dimensional problems.

5.4 Differential Constraints

Robot motions must usually conform to both global and local constraints. Global constraints on \mathcal{C} have been considered in the form of obstacles and possibly joint limits. Local constraints are modeled with differential equations, and are therefore called *differential constraints*. These limit the velocities, and possibly accelerations, at every point due to kinematic considerations, such as wheels in contact, and dynamical considerations, such as the conservation of angular momentum.

5.4.1 Concepts and Terminology

Let $\dot{\mathbf{q}}$ denote a velocity vector. Differential constraints on \mathcal{C} can be expressed either *implicitly* in the form $g_i(\mathbf{q}, \dot{\mathbf{q}}) = 0$ or *parametrically* in the form $\dot{\mathbf{x}} = f(\mathbf{q}, \mathbf{u})$. The implicit form is more general but often more difficult to understand and utilize. In the parametric form, a vector-valued equation indicates the velocity that is obtained for a given \mathbf{q} and \mathbf{u} , in which \mathbf{u} is an *input*, chosen from some *input space*, U . Let T denote an interval of time, starting at $t = 0$.

To model dynamics, the concepts are extended into a *phase space* X of the C-space. Usually each point $\mathbf{x} \in X$ represents both a configuration and velocity, $\mathbf{x} = (\mathbf{q}, \dot{\mathbf{q}})$. Both implicit and parametric representations are possible, yielding $g_i(\mathbf{x}, \dot{\mathbf{x}}) = 0$ and $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$,

respectively. The latter is a common *control system* definition. Note that $\dot{\mathbf{x}} = (\dot{\mathbf{q}}, \ddot{\mathbf{q}})$, which implies that acceleration constraints and full system dynamics can be expressed.

Planning in the state space X could lead to a straightforward definition of X_{obs} by declaring $\mathbf{x} \in X_{\text{obs}}$ if and only if $\mathbf{q} \in \mathcal{C}_{\text{obs}}$ for $\mathbf{x} = (\mathbf{q}, \dot{\mathbf{q}})$. However, another interesting possibility exists which provides some intuition about the difficulty of planning with dynamics. This possibility is based on the notion of a *region of inevitable collision*, which is defined as:

$$X_{\text{ric}} = \{ \mathbf{x}(0) \in X \mid \text{for any } \tilde{\mathbf{u}} \in \mathcal{U}_{\infty}, \exists t > 0 \text{ such that } \mathbf{x}(t) \in X_{\text{obs}} \}, \quad (5.4)$$

in which $\mathbf{x}(t)$ is the state at time t obtained by integrating the control function $\tilde{\mathbf{u}} : T \rightarrow U$ from $\mathbf{x}(0)$. The set \mathcal{U}_{∞} is a predefined set of all possible control functions. X_{ric} denotes the set of states in which the robot is either in collision or, because of momentum, it cannot do anything to avoid collision. It can be considered as an invisible obstacle region that grows with speed.

Under the general heading of *planning under differential constraints*, there are many important categories of problems that have received considerable attention in research literature. The term *nonholonomic planning*

was introduced for wheeled mobile robots [5.61]. A simple example is that a car cannot move sideways, thereby making parallel parking more difficult. In general, a *non-holonomic constraint* is a differential equality constraint that cannot be integrated into a constraint that involves no derivatives. Typically, nonholonomic constraints that appear in robotics are *kinematic*, and arise from wheels in contact [5.62]. Nonholonomic constraints may also arise from dynamics.

If a planning problem involves constraints on at least velocity and acceleration, the problem is often referred to as *kinodynamic planning* [5.63]. Usually, the model expresses a *fully actuated system*, which means that it can be expressed as $\ddot{\mathbf{q}} = \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{u})$, in which U contains an open set that includes the origin of \mathbb{R}^n (here, n is the dimension of both U and \mathcal{C}). It is possible for a problem to be nonholonomic, kinodynamic, both, or neither; however, in recent times, the terms are not used with much precision.

Trajectory planning is another important term, which has referred mainly to the problem of determining both a path and velocity function for a robot arm (e.g., PUMA 560). In the treatment below, all of these will be referred to as *planning under differential constraints*.

5.4.2 Discretization of Constraints

The only known methods for complete and optimal planning under differential constraints in the presence of obstacles are for the double integrator system with $X = \mathbb{R}$ [5.64] and $X = \mathbb{R}^2$ [5.65]. To develop planning algorithms in this context, several discretizations are often needed. For ordinary motion planning, only \mathcal{C} needed to be discretized; with differential constraints, T and possibly also U require discretization, in addition to \mathcal{C} (or X).

Discretization of the differential constraints is one of the most important issues. To solve challenging planning problems efficiently, it is often necessary to define *motion primitives* for the particular dynamical system [5.40, 66, 67]. One of the simplest ways to discretize the differential constraints is to construct a *discrete-time model*, which is characterized by three aspects:

1. The time interval T is partitioned into intervals of length Δt . This enables stages to be assigned, in which stage k indicates that $(k-1)\Delta t$ time has elapsed.
2. A finite subset U_d of the action space U is chosen. If U is already finite, then this selection may be $U_d = U$.

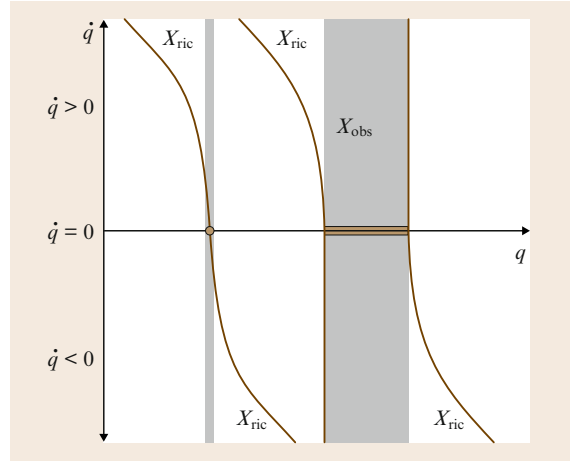


Fig. 5.11 The region of inevitable collision grows quadratically with the speed

3. The action $\mathbf{u}(t)$ must remain constant over each time interval.

From an initial state, \mathbf{x} , a *reachability tree* can be formed by applying all sequences of discretized actions. Figure 5.12 shows the path of this tree for the *Dubins car*, which is a kinematic model of a car that drives in the plane at unit speed and cannot move in reverse. The edges of the tree are circular arcs and line segments. For general systems, each trajectory segment in the tree is determined by numerical integration of $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u})$ for a given \mathbf{u} . In general, this can be viewed as an *in-*

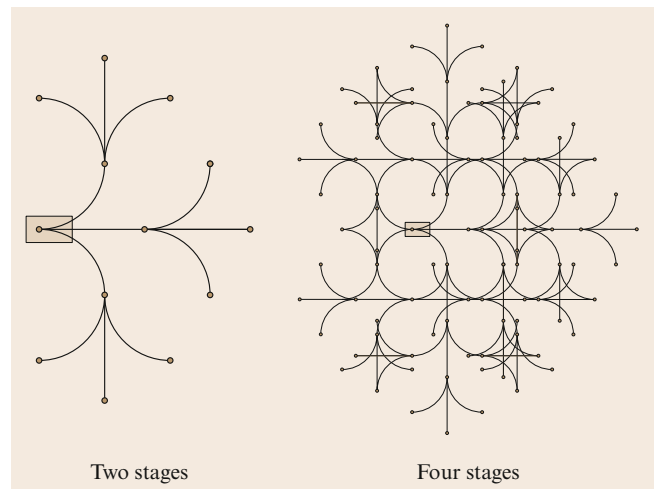


Fig. 5.12 A reachability tree for the Dubins car with three actions. The k -th stage produces 3^k new vertices

cremental simulator that takes an input and produces a trajectory segment according to $\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$.

5.4.3 Decoupled Approach

A popular paradigm for trajectory planning and other problems that involve dynamics is to decouple the problem into first planning a path and then computing a timing function along the path by performing a search in the space spanned by (s, \dot{s}) , in which s is the path parameter and \dot{s} is its first derivative. This leads to a diagram such as the one shown in Fig. 5.13, in which the upper region S_{obs} must be avoided because the corresponding motion of the mechanical system violates the differential constraints. Most methods are based on early work in [5.68, 69], and determine a *bang-bang-control*, which means that they switch between accelerating and decelerating at full speed. This applies to determining

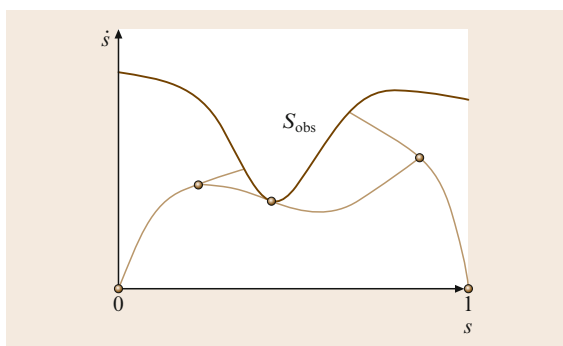


Fig. 5.13 An illustration of the bang-bang approach to computing a time-optimal trajectory. The solution trajectory is obtained by connecting the dots

time-optimal trajectories (optimal once constrained to the path). Dynamic programming can be used for more general problems [5.70].

For some problems and nonholonomic systems, *steering methods* have been developed to solve the two-point boundary value problem efficiently [5.62, 71]. This means that, for any pair of states, a trajectory that ignores obstacles but satisfies the differential constraints can be obtained. Moreover, for some systems, the complete set of optimal trajectories has been characterized [5.72, 73]. These control-based approaches enable straightforward adaptation of the sampling-based roadmap approach [5.74, 75]. One decoupled approach is to first plan a path that ignores differential constraints, and then incrementally transform it into one that obeys the constraints [5.62, 76].

5.4.4 Kinodynamic Planning

Due to the great difficulty of planning under differential constraints, many successful planning algorithms that address kinodynamic problems directly in the phase space X are sampling based.

Sampling-based planning algorithms proceed by exploring one or more reachability trees. Many parallels can be drawn with searching on a grid; however, reachability trees are more complicated because they do not necessarily involve a regular lattice structure. The vertex set of reachability trees is dense in most cases. It is therefore not clear how to search a bounded region exhaustively at a fixed resolution. It is also difficult to design approaches that behave like a multiresolution grid, in which refinements can be made arbitrarily to ensure resolution completeness.

Many algorithms attempt to convert the reachability tree into a lattice. This is the basis of the original kinodynamic planning work [5.63], in which the discrete-time approximation to the double integrator, $\ddot{\mathbf{q}} = \mathbf{u}$, is forced onto a lattice as shown in Fig. 5.14. This enables an approximation algorithm to be developed that solves the kinodynamic planning problem in time polynomial in the approximation quality $1/\epsilon$ and the number of primitives that define the obstacles. Generalizations of the methods to fully actuated systems are described in [5.7]. Surprisingly, it is even possible to obtain a lattice for some underactuated, nonholonomic systems [5.77].

If the reachability tree does not form a lattice, then one approach is to force it to behave as a lattice by imposing a regular cell decomposition over X (or \mathcal{C}), and allowing no more than one vertex per cell to be expanded in the reachability graph; see Fig. 5.15. This idea was

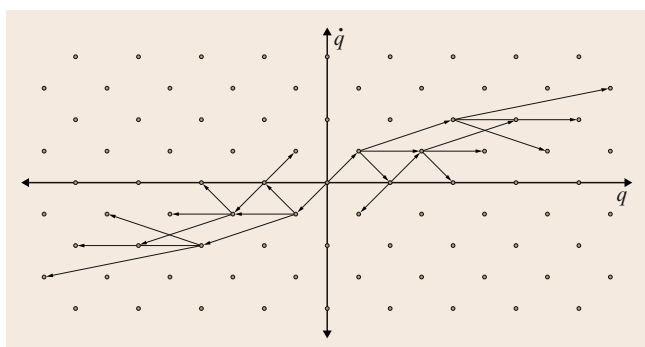


Fig. 5.14 Reachability graph from the origin, shown after three stages (the true edges are actually parabolic arcs when acceleration or deceleration occurs). Note that a lattice is obtained, but the distance traveled in one stage increases as $|\dot{q}|$ increases

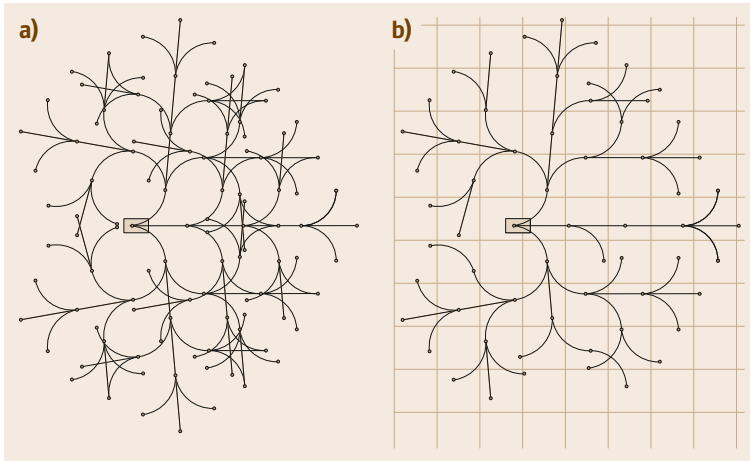


Fig. 5.15 (a) The first four stages of a dense reachability graph for the Dubins car; (b) one possible search graph, obtained by allowing at most one vertex per cell. Many branches are pruned away. In this simple example there are no cell divisions along the θ axis

introduced in [5.78]. In their version of this approach, the reachability graph is expanded by dynamic programming. Each cell is initially marked as being in collision or being collision free, but not yet visited. As cells are visited during the search, they become marked as such. If a potential new vertex lands in a visited cell, it is not saved. This has the effect of pruning the reachability tree.

Other related approaches do not try to force the reachability tree onto a lattice. **RRTs** were designed to expand the tree in a way that is biased toward

covering as much new territory as possible in each iteration [5.79]. Planners that are based on the concept of expansive trees attempt to control the density of vertices in the tree by analyzing neighborhoods [5.44]. The path-directed subdivision tree planner expands a tree, while building an adaptive subdivision of the state space, so as to avoid resampling the same regions of the space [5.47, 80]. Such approaches can be biased to accelerate the expansion of the tree towards a goal, while still providing the weaker probabilistic completeness guarantee [5.48].

5.5 Extensions and Variations

A brief overview of other important extensions to the basic motion planning problem are presented in this section.

5.5.1 Closed Kinematic Chains

In many cases, the robot may consist of links that form closed loops. This arises in many important applications, for example, if two arms grasp an object then a loop is formed and a humanoid robot forms a loop if both legs touch the ground. For *parallel robots*, loops are intentionally designed into the robot [5.81]; a classic example is the Stewart–Gough platform. To model closed-chain problems, the loops are broken so that a kinematic tree of links is obtained. The main complication is that constraints on \mathcal{C} of the form $h(\mathbf{q}) = 0$ are introduced, which require that the loops are maintained. This causes great trouble for most planning algorithms

because without loops a parameterization of \mathcal{C} was available. The closure constraints restrict the planning to a lower-dimensional subset of \mathcal{C} for which no parameterization is given. Computing a parameterization is generally difficult or impossible [5.82], although there has been recent progress for some special cases [5.83].

Sampling-based approaches can generally be adapted to handle closed chains. The main difficulty is that the samples $\alpha(i)$ over \mathcal{C} are unlikely to be configurations that satisfy closure. In [5.84], both **RRTs** and **PRMs** were adapted to closed chains. **RRTs** performed much better because a costly optimization was required in the **PRM** to move samples onto the closure subspace; **RRTs** on the other hand do not require samples to lie in this subspace. By decomposing chains into active and passive links, followed by inverse kinematics computations, performance was dramatically improved for **PRMs** in [5.85]. This idea was further improved by the introduction of

the *random loop generator* (RLG). Based on this, some of the most challenging closed-chain planning problems ever solved appear in [5.86].

5.5.2 Manipulation Planning

In most forms of motion planning, the robot is not allowed to touch obstacles. Suppose instead that it is expected to interact with its environment by manipulating objects. The goal may be to bring an object from one place to another, or to rearrange a collection of objects. This leads to a kind of hybrid motion planning problem, which mixes discrete and continuous spaces. There are discrete modes that correspond to whether the robot is carrying a part [5.87]. In the *transit mode*, the robot moves toward a part. In the *transfer mode*, it carries the part. Transitions between modes require meeting specific grasping and stability requirement. One important variant of manipulation planning is *assembly planning*, in which the goal is to fit a collection of pieces together to make an assembled product [5.88]. Most motion planning work makes limiting assumptions on the kinds of interaction that can occur between the robot and the objects. For richer models of manipulation, see [5.89].

5.5.3 Time-Varying Problems

Suppose that the workspace contains moving obstacles whose trajectories are specified as a function of time. Let $T \subset \mathbb{R}$ denote the *time interval*, which may be *bounded*

or *unbounded*. A state X is defined as $X = \mathcal{C} \times T$, in which \mathcal{C} is the usual C-space of the robot. The obstacle region in X is characterized as

$$X_{\text{obs}} = \{(q, t) \in X \mid \mathcal{A}(q) \cap \mathcal{O}(t) \neq \emptyset\}, \quad (5.5)$$

in which $\mathcal{O}(t)$ is a time-varying obstacle. Many planning algorithms can be adapted to X , which has only one more dimension than \mathcal{C} . The main complication is that time must always increase along a path through X .

For the easiest version of the problem, there is no bound on the robot speed. In this case, virtually any sampling-based algorithm can be adapted. Incremental searching and sampling methods apply with little modification, except that paths are directed so that forward time progress is made. Using bidirectional approaches is more difficult for time-varying problems because the goal is usually not a single point due to the time dependency. Sampling-based roadmaps can be adapted; however, a *directed roadmap* is needed, in which every edge must be directed to yield a time-monotonic path.

If the motion model is *algebraic* (i. e., expressed with polynomials) then X_{obs} is semi-algebraic. This enables cylindrical algebraic decomposition to apply. If X_{obs} is polyhedral, as depicted in Fig. 5.16, then vertical decomposition can be used. It is best to first sweep the plane along the T -axis, stopping at the critical times when the linear motion changes.

There has been no consideration so far of the speed at which the robot must move to avoid obstacles. It is obviously impractical in many applications if the solution requires the robot to move arbitrarily fast. One step towards making a realistic model is to enforce a bound on the speed of the robot. Unfortunately, the problem is considerably more difficult. Even for piecewise-linear motions of obstacles in the plane, the problem has been established to be PSPACE-hard [5.90]. A complete algorithm based on the shortest-path roadmap is presented in [5.91].

An alternative to defining the problem in $\mathcal{C} \times T$ is to decouple it into a *path planning* part and a *motion timing* part. A collision-free path in the absence of obstacles is first computed. A search in a two-dimensional space is then performed to determine the *timing function* (or *time scaling*) for the path.

5.5.4 Multiple Robots

A simple extension to the basic motion planning problem can be made to handle multibody robots by including robot self-intersections; however, it is important to specify the pairs of bodies for which collision is unac-

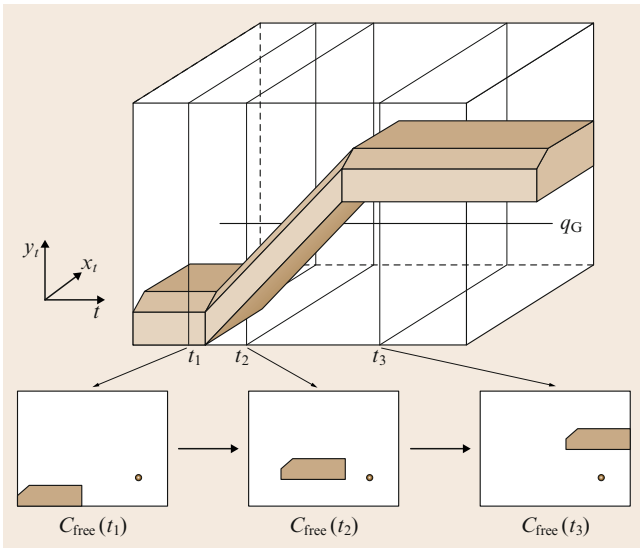


Fig. 5.16 A time-varying example with linear obstacle motion.

ceptable, for example, consecutive links in a robot arm are allowed to touch.

Substantial attention has been devoted to the problem of planning for multiple robots. Suppose there are m robots. A state space is defined that considers the configurations of all robots simultaneously,

$$X = \mathcal{C}^1 \times \mathcal{C}^2 \times \dots \times \mathcal{C}^m. \quad (5.6)$$

A state $\mathbf{x} \in X$ specifies all robot configurations, and may be expressed as $\mathbf{x} = (\mathbf{q}^1, \mathbf{q}^2, \dots, \mathbf{q}^m)$. The dimension of X is N , which is $N = \sum_{i=1}^m \dim(\mathcal{C}^i)$.

There are two sources of obstacle regions in the state space: (1) *robot–obstacle* collisions, and (2) *robot–robot* collisions. For each i such that $1 \leq i \leq m$, the subset of X that corresponds to robot \mathcal{A}^i in collision with the obstacle region \mathcal{O} is

$$X_{\text{obs}}^i = \{\mathbf{x} \in X \mid \mathcal{A}^i(\mathbf{q}^i) \cap \mathcal{O} \neq \emptyset\}. \quad (5.7)$$

This models the robot–obstacle collisions.

For each pair, \mathcal{A}^i and \mathcal{A}^j , of robots, the subset of X that corresponds to \mathcal{A}^i in collision with \mathcal{A}^j is

$$X_{\text{obs}}^{ij} = \{\mathbf{x} \in X \mid \mathcal{A}^i(\mathbf{q}^i) \cap \mathcal{A}^j(\mathbf{q}^j) \neq \emptyset\}. \quad (5.8)$$

Both (5.7) and (5.8) will be combined in (5.9) to yield X_{obs} . The obstacle region in X is

$$X_{\text{obs}} = \left(\bigcup_{i=1}^m X_{\text{obs}}^i \right) \cup \left(\bigcup_{ij, i \neq j} X_{\text{obs}}^{ij} \right). \quad (5.9)$$

Once these definitions have been made, any general-purpose planning algorithm can be applied because X and X_{obs} appear no different from \mathcal{C} and \mathcal{C}_{obs} , except that the dimension N may be very high. Approaches that plan directly in X are called *centralized*. The high dimensionality of X motivates the development of *decoupled* approaches that handle some aspects of the planning independently for each robot. Decoupled approaches are usually more efficient, but this usually comes at the expense of sacrificing completeness. An early decoupled approach is *prioritized planning* [5.92, 93], in which a path and timing function is computed for the i -th robot while treating the first $i - 1$ robots as moving obstacles as they follow their paths. Another decoupled approach is *fixed-path* coordination [5.94], in which the paths are planned independently for each robot, and then their timing functions are determined by computing a collision-free path through an m -dimensional *coordination space*. Each axis in this space corresponds to the domain of the path of one robot. Figure 5.17 shows an example. The idea has been generalized to coordination on roadmaps [5.95, 96].

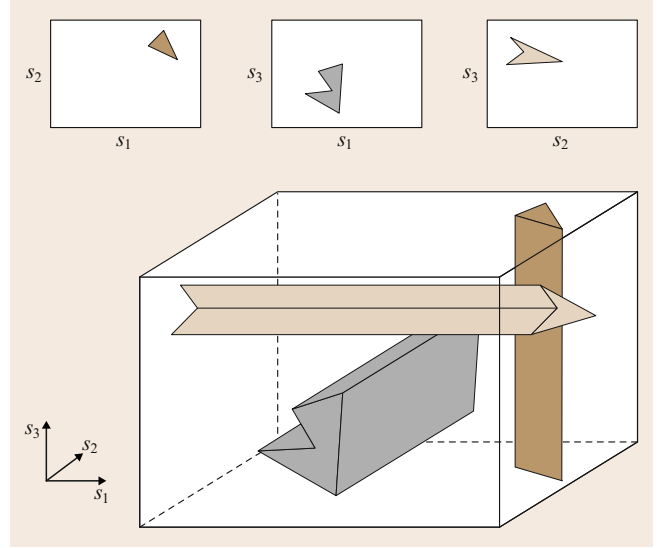


Fig. 5.17 The obstacles that arise from coordinating m robots are always cylindrical. The set of all $\frac{1}{2}m(m-1)$ axis-aligned 2-D projections completely characterizes X_{obs}

5.5.5 Uncertainty in Predictability

If the execution of the plan is not predictable, then feedback is needed. The uncertainty may be modeled either *implicitly*, which means that the plan is able to respond to unexpected future configurations, or *explicitly*, which means that the uncertainty is precisely characterized and analyzed in the development of a plan. Potential-function-based approaches are one way of achieving feedback motion planning.

A plan can be represented as a vector field over $\mathcal{C}_{\text{free}}$, in which each vector indicates the required velocity. The integral curves of the field should flow into the goal without leaving \mathcal{C}_{obs} . If dynamics are a concern, then the vector field can be tracked by an *acceleration-based control model*:

$$\mathbf{u} = K(f(\mathbf{q}) - \dot{\mathbf{q}}) + \nabla_{\dot{\mathbf{q}}} f(\mathbf{q}) \quad (5.10)$$

in which K is a scalar *gain constant*. Alternatively, a vector field may be designed directly on the phase space, X ; however, there are not methods to compute such fields efficiently under general conditions. This can also be considered as a feedback control problem with implicit, nonlinear constraints on X .

If the uncertainty is modeled explicitly, then a *game against nature* is obtained, in which the uncertainty is caused by a special decision maker called *nature*. The decisions of nature can either be modeled *non-*

deterministically, which means that a set of possible actions is specified, or *probabilistically*, which means that a probability distribution or density is specified over the nature actions. Under nondeterministic uncertainty, *worst-case analysis* is usually performed to select a plan; under probabilistic uncertainty, *expected-case analysis* is usually performed. Numerous approaches exist for such problems, including value iteration, Dijkstra-like algorithms, and reinforcement learning algorithms [5.7].

5.5.6 Sensing Uncertainty

Consider solving tasks such as localization, map building, manipulation, target tracking, and pursuit evasion (hide-and-seek) with limited sensing. If the current configuration or state is not known during execution, then the problem is substantially more difficult. Information is obtained from sensors, and the problem naturally

lives in an *information space* or *I-space* (see Chap. 11 of [5.7]). The state may include the configuration, velocities, or even the map of the environment (e.g., obstacles). The most basic I-space is the set of all histories that can be obtained during execution, based on all sensing observations, actions previously applied, and the initial conditions. The goal in developing efficient algorithms in this context is to determine information mappings that reduce the I-space size or complexity so that plans that use *information feedback* can be computed. The traditional way to use the information state is for estimating the state. This is sufficient for solving many tasks, but it is often not necessary. It may be possible to design and execute successful plans without ever knowing the current state. This can lead to more robust robot systems which may also be cheaper to manufacture due to weaker sensing requirements. For more material related to this topic, see any of the chapters in Part C of this book.

5.6 Advanced Issues

We cover here a series of more advanced issues, such as topics from topology and sampling theory, and how they influence the performance of motion planners. The last section is devoted to computational algebraic geometry techniques that achieve completeness in the general case. Rather than being a practical alternative, these techniques serve as an upper bound on the best asymptotic running time that could be obtained.

5.6.1 Topology of Configuration Spaces

Manifolds

One reason that the topology of a C-space is important is because it affects its representation. Another reason is that, if a path-planning algorithm can solve problems in a topological space, then that algorithm may carry over to topologically equivalent spaces.

The following definitions are important in order to describe the topology of C-space. A map $\phi : S \rightarrow T$ is called a *homeomorphism* if ϕ is a bijection and both ϕ and ϕ^{-1} are continuous. When such a map exists, S and T are said to be *homeomorphic*. A set S is an *n-dimensional manifold* if it is locally homeomorphic to \mathbb{R}^n , meaning that each point in S possesses a neighborhood that is homeomorphic to \mathbb{R}^n . For more details, see [5.97, 98].

In the vast majority of motion planning problems, the configuration space is a manifold. An example of a C-space that is not a manifold is the closed unit square: $[0, 1] \times [0, 1] \subset \mathbb{R}^2$, which is a manifold with boundary obtained by pasting the one-dimensional boundary on the two-dimensional open set $(0, 1) \times (0, 1)$. When a C-space is a manifold, then we can represent it with just n parameters, in which n is the dimension of the configuration space. Although an n -dimensional manifold can be represented using as few as n parameters, due to constraints it might be easier to use a representation that has higher number of parameters, e.g., the unit circle \mathbb{S}^1 can be represented as $\mathbb{S}^1 = \{(x, y) | x^2 + y^2 = 1\}$ by embedding \mathbb{S}^1 in \mathbb{R}^2 . Similarly, the torus T^2 can be embedded in \mathbb{R}^3 .

Representation

Embeddings into higher-dimensional spaces can facilitate many C-space operations. For example, the orientation of a rigid body in space can be represented by a $n \times n$ matrix of real numbers. The n^2 matrix entries must satisfy a number of smooth equality constraints, making the manifold of such matrices a *submanifold* of \mathbb{R}^{m^2} . One advantage is that these matrices can be multiplied to get another matrix in the manifold. For example, the orientation of a rigid-body in n -dimensional space ($n = 2$ or 3) is described by the set $SO(n)$, the set

of all $n \times n$ rotation matrices. The position and orientation of a rigid body is represented by the set $SE(n)$, the set of all $n \times n$ homogeneous transformation matrices. These matrix groups can be used to (1) represent rigid-body configurations, (2) change the reference frame for the representation of a configuration, and (3) displace a configuration.

There are numerous parameterizations of $SO(3)$ [5.99] but unit quaternions correctly preserve the \mathcal{C} -space topology as \mathbb{S}^1 represents 2-D rotations. Quaternions were introduced in Chap. 1. There is, however, a two-to-one correspondence between unit quaternions and 3-D rotation matrices. This causes a topological issue that is similar to the equivalence of 0 and 2π for 2-D rotations. One way to account for this is to declare antipodal (opposite) points on \mathbb{S}^3 to be equivalent. In planning, only the upper hemisphere of \mathbb{S}^3 is needed, and paths that cross the equator instantly reappear on the opposite side of \mathbb{S}^3 , heading back into the northern hemisphere. In topology, this is called a real projective space: \mathbb{RP}^3 . Hence, the C-space of a 3-D body capable only of rotation is \mathbb{RP}^3 . If both translation and rotation are allowed, then $SE(3)$, the set of all 4×4 homogeneous transformation matrices, yields:

$$\mathcal{C} = \mathbb{R}^3 \times \mathbb{RP}^3, \quad (5.11)$$

which is six dimensional. A configuration $\mathbf{q} \in \mathcal{C}$ can be expressed using quaternions with seven coordinates, (x, y, z, a, b, c, d) , in which $a^2 + b^2 + c^2 + d^2 = 1$.

5.6.2 Sampling Theory

Since the most successful paradigm for motion planning today is the sampling-based framework, presented in Sect. 5.2, sampling theory becomes relevant to the motion planning problem.

Metrics in Configuration/State Spaces

Virtually all sampling-based methods require some notion of distance on \mathcal{C} . For example, the sampling-based roadmap method selects candidate vertices to connect a new configuration given a distance-defined neighborhood. Similarly, the rapidly exploring dense trees expands the tree from the nearest node of the tree to a newly sampled configuration. Usually, a *metric*, $\rho: \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}$, is defined, which satisfies the standard axioms: nonnegativity, reflexivity, symmetry, and the triangle inequality.

Two difficult issues that arise in constructing a metric are: (1) the topology of \mathcal{C} must be respected, and (2) several different quantities, such as linear and angu-

lar displacements, must be compared in some way. To illustrate the second issue, consider defining a metric ρ_z for a space constructed as $Z = X \times Y$ as

$$\begin{aligned} \rho_z(z, z') &= \rho_z(x, y, x', y') \\ &= c_1 \rho_x(x, x') + c_2 \rho_y(y, y'). \end{aligned} \quad (5.12)$$

Above, c_1 and c_2 are arbitrary positive constants that indicate the relative weights of the two components. For a 2-D rotation, θ_i , expressed as $a_i = \cos \theta_i$ and $b_i = \sin \theta_i$, a useful metric is:

$$\rho(a_1, b_1, a_2, b_2) = \cos^{-1}(a_1 a_2 + b_1 b_2). \quad (5.13)$$

The 3-D equivalent is obtained by defining

$$\rho_0(\mathbf{h}_1, \mathbf{h}_2) = \cos^{-1}(a_1 a_2 + b_1 b_2 + c_1 c_2 + d_1 d_2), \quad (5.14)$$

in which each $\mathbf{h}_i = (a_i, b_i, c_i, d_i)$ is a unit quaternion. The metric is defined as $\rho(\mathbf{h}_1, \mathbf{h}_2) = \min(\rho_0(\mathbf{h}_1, \mathbf{h}_2), \rho_0(\mathbf{h}_1, -\mathbf{h}_2))$, by respecting the required identification of antipodal points. This computes the shortest distance in \mathbb{R}^4 , for a path constrained to the unit sphere.

In some algorithms, defining volume on \mathcal{C} may also be important. In general, this leads to a *measure space*, for which the volume function (called the *measure*) must satisfy axioms that resemble the probability axioms, but without normalization. For transformation groups, one must be careful to define volumes in a way that is invariant with respect to transformations. Such volumes are called *Haar measures*. Defining volumes via balls using the metric definitions (5.13) and (5.14) actually satisfy this concern.

Probabilistic versus Deterministic Sampling

The C-space may be sampled *probabilistically* or *deterministically*. Either way, the requirement is usually that a dense sequence α of samples is obtained. This means that, in the limit as the number of samples tends to infinity, the samples become arbitrarily close to every point in \mathcal{C} . For probabilistic sampling, this denseness (with probability one) ensures *probabilistic completeness* of a planning algorithm. For deterministic sampling, it ensures *resolution completeness*, which means that, if a solution exists, the algorithm is guaranteed to find it; otherwise, it may run forever.

For probabilistic sampling, samples are selected randomly over \mathcal{C} , using a uniform probability density function. To obtain uniformity in a meaningful way, the Haar measure should be used. This is straightforward in many cases; $SO(3)$ however is tricky. A uniform

(with respect to Haar measure) random quaternion is selected as follows. Choose three points $u_1, u_2, u_3 \in [0, 1]$ uniformly at random, and let [5.100]

$$\mathbf{h} = \left(\sqrt{1-u_1} \sin 2\pi u_2, \sqrt{1-u_1} \cos 2\pi u_2, \sqrt{u_1} \sin 2\pi u_3, \sqrt{u_1} \cos 2\pi u_3 \right). \quad (5.15)$$

Even though random samples are uniform in some sense, they are also required to have some irregularity to satisfy statistical tests. This has motivated the development of deterministic sampling schemes that offer better performance [5.101]. Instead of being concerned with randomness, deterministic sampling techniques are designed to optimize criteria, such as *discrepancy* and *dispersion*. Discrepancy penalizes regularity in the sample, which frequently causes trouble in numerical integration. Dispersion gives the radius of the largest empty (not containing samples) ball. Thus, driving dispersion down quickly means that the whole space is explored quickly. Deterministic samples may be *irregular* neighborhood structure (appearing much like random samples), or *regular* neighborhood structure, which means that points are arranged along a grid or lattice. For more details in the context of motion planning, see [5.7].

5.6.3 Computational Algebraic Geometry Techniques

Sampling-based algorithms provide good practical performance at the expense of achieving only a weaker form of completeness. On the other hand, complete al-

gorithms, which are the focus of this section, are able to deduce that there is no solution to a planning problem.

Complete algorithms are able to solve virtually any motion planning problem as long as \mathcal{C}_{obs} is represented by patches of algebraic surfaces. Formally, the model must be *semi-algebraic*, which means that it is formed from unions and intersections of roots of multivariate polynomials in q , and for computability, the polynomials must have rational coefficients (otherwise roots may not have finite representations). The set of all roots to polynomials with rational coefficients is called *real algebraic numbers* and has many nice computational properties. See [5.12, 102–104] for more information on the exact representation and calculation with real algebraic numbers. For a gentle introduction to algebraic geometry, see [5.82].

To use techniques based on algebraic geometry, the first step is to convert the models into the required polynomials. Suppose that the models, the robot, \mathcal{A} , and the obstacles \mathcal{O} are semi-algebraic (this includes polyhedral models). For any number of attached 2-D or 3-D bodies, the kinematic transformations can be expressed using polynomials. Since polynomial transformations of polynomials yield polynomials, the transformed robot model is polynomial. The algebraic surfaces that comprise \mathcal{C}_{obs} are computed by carefully considering all contact types, which characterize all ways to pair a robot feature (faces, edges, vertices) with an obstacle feature [5.6, 7, 9, 105]. This step already produces too many model primitives to be useful in most applications.

Once the semi-algebraic representation has been obtained, powerful techniques from algebraic geometry can be exploited. One of the most widely known algorithms, *cylindrical algebraic decomposition* [5.102, 106, 107], provides the information needed to solve the motion planning problem. It was originally designed to determine whether *Tarski sentences*, which involve quantifiers and polynomials, are satisfiable, and to find an equivalent expression that does not involve quantifiers. The decomposition produces a finite set of cells over which the signs of the polynomials remain fixed. This enables a systematic approach to satisfiability and quantifier elimination. It was recognized by Schwartz and Sharir [5.104] that it also solves motion planning.

The method is conceptually simple, but there are many difficult technical details. The decomposition is called cylindrical because the cells are organized into vertical columns of cells, see Fig. 5.18 for a 2-D example. There are two kinds of critical events, shown in Fig. 5.19. At critical points, rays are extended indef-

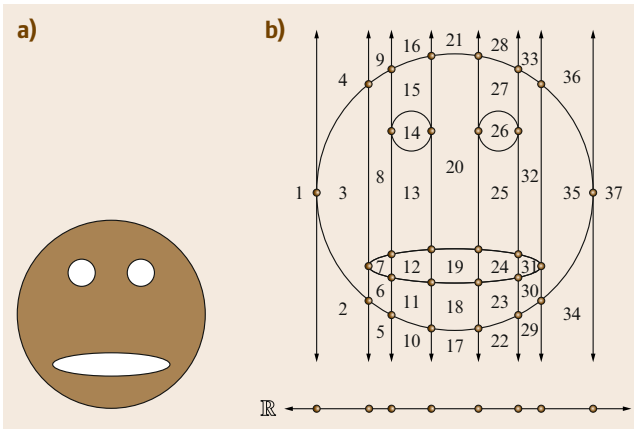


Fig. 5.18 (a) A face modeled with four algebraic primitives, and (b) a cylindrical algebraic decomposition of the face

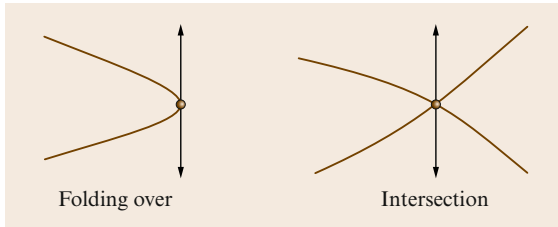


Fig. 5.19 Critical points occur either when the surface folds over in the vertical direction or when surfaces intersect

initely in both vertical directions. The decomposition differs from the vertical decomposition in Fig. 5.7 because there the rays were only extended until the next obstacle was hit. Here, columns of cells are obtained.

In n dimensions, each column represents a chain of cells. The first and last cells are n -dimensional and unbounded. The remaining cells are bounded and alternate between being $(n - 1)$ -dimensional and n dimensional. The bounded n -dimensional cells are bounded above and below by the roots of single multivariate polynomials. This makes it simple to describe the cells and their connectivity. To compute this cell decomposition, the algorithm constructs a cascading chain of projections. In the first step, \mathcal{C}_{obs} is projected from \mathbb{R}^n to \mathbb{R}^{n-1} . This is followed by a projection into \mathbb{R}^{n-2} . This repeats until \mathbb{R} is obtained with a univariate polynomial that encodes the places at which all critical boundaries need to be placed. In a second phase of the algorithm, a series of liftings is performed. Each lifting takes the polynomials and cell decomposition over \mathbb{R}^i and lifts them via columns of cells to \mathbb{R}^{i+1} . A single lifting is illustrated in Fig. 5.18b. The running time of the full algorithm depends on the particular methods used to perform the algebraic computations. The total running time required to use cylindrical algebraic decomposition for motion planning is bounded by $(md)^{O(1)^n}$, in which m is the number of polynomials to describe \mathcal{C}_{obs} (a huge number), and d is the maximum algebraic degree. [It may seem odd for $O(\cdot)$ to appear

in the middle of an expression. In this context, it means that there exists some $c \in [0, \infty)$ such that the running time is bounded by $(md)^{c^n}$. Note that another O is not necessary in the front of the whole formula.] The main point to remember is that the algorithm is doubly exponential in the dimension of \mathcal{C} (even the number of cells is doubly exponential).

Although performing the cylindrical decomposition is sufficient for solving motion planning, it computes more information than is necessary. This motivates Canny's roadmap algorithm [5.12], which produces a roadmap directly from the semi-algebraic set, rather than constructing a cell decomposition along the way. Since there are doubly exponentially many cells in the cylindrical algebraic decomposition, avoiding this construction pays off. The resulting roadmap method of Canny solves the motion planning problem in time that is again polynomial in the number of polynomials and polynomial in the algebraic degree, but is only singly exponential in dimension [5.12].

The basic idea is to find silhouette curves in \mathbb{R}^2 of \mathcal{C}_{obs} in \mathbb{R}^n . The method finds zero-dimensional critical points and one-dimensional critical curves. The critical curves become roadmap edges, and the critical points are places at which the algorithm recursively finds silhouettes of $(n - 1)$ -dimensional slices of \mathcal{C}_{obs} . These contribute more critical points and curves. The curves are added to the roadmap, and the algorithm recurses again on the critical points. The recursive iterations terminate at $n = 2$. Canny showed that the resulting union of critical curves preserves the connectivity of \mathcal{C}_{obs} (and hence, $\mathcal{C}_{\text{free}}$). Some of the technical issues are: the algorithm works with a stratification of \mathcal{C}_{obs} into manifolds; there are strong general position assumptions that are hard to meet; paths are actually considered along the boundary of $\mathcal{C}_{\text{free}}$; and the method does not produce a parameterized solution path. For improvements to Canny's algorithm and many other important details, see [5.102].

5.7 Conclusions and Further Reading

The brief survey given here hardly does justice to motion planning, which is a rich and active research field. For more details, we recommend consulting two recent textbooks [5.5, 7]. In addition, see the classic textbook

of Latombe [5.6], the classic papers in [5.4], and the recent surveys in [5.2, 3]. Furthermore, consult the related handbook chapters that were indicated throughout this chapter.

References

- 5.1 J.H. Reif: Complexity of the mover's problem and generalizations, IEEE Symp. Found. Comput. Sci. (1979) pp. 421–427
- 5.2 H.H. Gonzalez-Banos, D. Hsu, J.C. Latombe: Motion planning: Recent developments. In: *Autonomous Mobile Robots: Sensing, Control, Decision-Making and Applications*, ed. by S.S. Ge, F.L. Lewis (CRC, Boca Raton 2006)
- 5.3 S.R. Lindemann, S.M. LaValle: Current issues in sampling-based motion planning. In: *Robotics Research: The Eleventh International Symposium*, ed. by P. Dario, R. Chatila (Springer, Berlin 2005) pp. 36–54
- 5.4 J.T. Schwartz, M. Sharir: A survey of motion planning and related geometric algorithms, Artif. Intell. J. **37**, 157–169 (1988)
- 5.5 H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun: *Principles of Robot Motion: Theory, Algorithms, and Implementations* (MIT Press, Cambridge 2005)
- 5.6 J.C. Latombe: *Robot Motion Planning* (Kluwer, Boston 1991)
- 5.7 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2006)
- 5.8 S. Udupa: Collision detection and avoidance in computer controlled manipulators. Ph.D. Thesis (Dept. of Electrical Engineering, California Institute of Technology 1977)
- 5.9 T. Lozano-Pérez: Spatial planning: A configuration space approach, IEEE Trans. Comput. **C-32**(2), 108–120 (1983)
- 5.10 J.T. Schwartz, M. Sharir: On the piano movers' problem: III. Coordinating the motion of several independent bodies, Int. J. Robot. Res. **2**(3), 97–140 (1983)
- 5.11 J.T. Schwartz, M. Sharir: On the piano movers' problem: V. The case of a rod moving in three-dimensional space amidst polyhedral obstacles, Commun. Pure Appl. Math. **37**, 815–848 (1984)
- 5.12 J.F. Canny: *The Complexity of Robot Motion Planning* (MIT Press, Cambridge 1988)
- 5.13 D. Halperin, M. Sharir: A near-quadratic algorithm for planning the motion of a polygon in a polygonal environment, Discrete Comput. Geom. **16**, 121–134 (1996)
- 5.14 J.E. Hopcroft, J.T. Schwartz, M. Sharir: On the complexity of motion planning for multiple independent objects: PSPACE-hardness of the warehouseman's problem, Int. J. Robot. Res. **3**(4), 76–88 (1984)
- 5.15 J. Canny, J. Reif: New lower bound techniques for robot motion planning problems, IEEE Symp. Found. Comput. Sci. (1987) pp. 49–60
- 5.16 M.C. Lin, J.F. Canny: Efficient algorithms for incremental distance computation, IEEE Int. Conf. Robot. Autom. (1991)
- 5.17 P. Jiménez, F. Thomas, C. Torras: Collision detection algorithms for motion planning. In: *Robot Motion Planning and Control*, ed. by J.P. Laumond (Springer, Berlin 1998) pp. 1–53
- 5.18 M.C. Lin, D. Manocha: Collision and proximity queries. In: *Handbook of Discrete and Computational Geometry, 2nd Ed*, ed. by J.E. Goodman, J. O'Rourke (Chapman Hall/CRC, New York 2004) pp. 787–807
- 5.19 L.E. Kavraki, P. Svestka, J.C. Latombe, M.H. Overmars: Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Autom. **12**(4), 566–580 (1996)
- 5.20 N.M. Amato, O.B. Bayazit, L.K. Dale, C. Jones, D. Vallejo: OBPRM: an obstacle-based PRM for 3D workspaces, Workshop Algorith. Found. Robot. (1998) pp. 155–168
- 5.21 V. Boor, M.H. Overmars, A.F. van der Stappen: The Gaussian sampling strategy for probabilistic roadmap planners, IEEE Int. Conf. Robot. Autom. (1999) pp. 1018–1023
- 5.22 C. Holleman, L.E. Kavraki: A framework for using the workspace medial axis in PRM planners, IEEE Int. Conf. Robot. Autom. (2000) pp. 1408–1413
- 5.23 J.M. Lien, S.L. Thomas, N.M. Amato: A general framework for sampling on the medial axis of the free space, IEEE Int. Conf. Robot. Autom. (2003)
- 5.24 S.M. LaValle, M.S. Branicky, S.R. Lindemann: On the relationship between classical grid search and probabilistic roadmaps, Int. J. Robot. Res. **23**(7/8), 673–692 (2004)
- 5.25 T. Siméon, J.-P. Laumond, C. Nissoux: Visibility based probabilistic roadmaps for motion planning, Adv. Robot. **14**(6), 477–493 (2000)
- 5.26 J. Barraquand, L. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, P. Raghavan: A random sampling scheme for robot path planning. In: *Proceedings International Symposium on Robotics Research*, ed. by G. Giralt, G. Hirzinger (Springer, New York 1996) pp. 249–264
- 5.27 A. Ladd, L.E. Kavraki: Measure theoretic analysis of probabilistic path planning, IEEE Trans. Robot. Autom. **20**(2), 229–242 (2004)
- 5.28 R. Geraerts, M. Overmars: Sampling techniques for probabilistic roadmap planners, Int. Conf. Intell. Auton. Syst. (2004)
- 5.29 D. Hsu, T. Jiang, J. Reif, Z. Sun: The bridge test for sampling narrow passages with probabilistic roadmap planners, IEEE Int. Conf. Robot. Autom. (2003)
- 5.30 R. Bohlin, L. Kavraki: Path planning using lazy PRM, IEEE Int. Conf. Robot. Autom. (2000)
- 5.31 B. Burns, O. Brock: Sampling-based motion planning using predictive models, IEEE/RSJ Int. Conf. Intell. Robot. Autom. (2005)

- 5.32 P. Ito: Constructing probabilistic roadmaps with powerful local planning and path optimization, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2002) pp. 2323–2328
- 5.33 P. Leven, S.A. Hutchinson: Using manipulability to bias sampling during the construction of probabilistic roadmaps, IEEE Trans. Robot. Autom. **19**(6), 1020–1026 (2003)
- 5.34 D. Nieuwenhuisen, M.H. Overmars: Useful cycles in probabilistic roadmap graphs, IEEE Int. Conf. Robot. Autom. (2004) pp. 446–452
- 5.35 S.M. LaValle, J.J. Kuffner: Rapidly-exploring random trees: progress and prospects. In: *Algorithmic and Computational Robotics: New Direction*, ed. by B.R. Donald, K.M. Lynch, D. Rus (A. K. Peters, Wellesley 2001) pp. 293–308
- 5.36 K.E. Bekris, B.Y. Chen, A. Ladd, E. Plaku, L.E. Kavraki: Multiple query probabilistic roadmap planning using single query primitives, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003)
- 5.37 M. Strandberg: Augmenting RRT-planners with local trees, IEEE Int. Conf. Robot. Autom. (2004) pp. 3258–3262
- 5.38 J. J. Kuffner, S. M. LaValle: An efficient approach to path planning using balanced bidirectional RRT search, Techn. Rep. CMU-RI-TR-05-34 Robotics Institute, Carnegie Mellon University, Pittsburgh (2005)
- 5.39 J. Bruce, M. Veloso: Real-time randomized path planning for robot navigation, IEEE/RSJ Int. Conf. Intell. Robot. Autom. (2002)
- 5.40 E. Frazzoli, M.A. Dahleh, E. Feron: Real-time motion planning for agile autonomous vehicles, AIAA J. Guid. Contr. **25**(1), 116–129 (2002)
- 5.41 M. Kallmann, M. Mataric: Motion planning using dynamic roadmaps, IEEE Int. Conf. Robot. Autom. (2004)
- 5.42 A. Yershova, L. Jaillet, T. Simeon, S.M. LaValle: Dynamic-domain RRTs: efficient exploration by controlling the sampling domain, IEEE Int. Conf. Robot. Autom. (2005)
- 5.43 D. Hsu, J.C. Latombe, R. Motwani: Path planning in expansive configuration spaces, Int. J. Comput. Geom. Appl. **4**, 495–512 (1999)
- 5.44 D. Hsu, R. Kindel, J.C. Latombe, S. Rock: Randomized kinodynamic motion planning with moving obstacles. In: *Algorithmic and Computational Robotics: New Directions*, ed. by B.R. Donald, K.M. Lynch, D. Rus (A.K. Peters, Wellesley 2001)
- 5.45 G. Sánchez, J.-C. Latombe: A single-query bi-directional probabilistic roadmap planner with lazy collision checking, ISRR Int. Symp. Robot. Res. (2001)
- 5.46 S. Carpin, G. Pillonetto: Robot motion planning using adaptive random walks, IEEE Int. Conf. Robot. Autom. (2003) pp. 3809–3814
- 5.47 A. Ladd, L.E. Kavraki: Fast exploration for robots with dynamics, Workshop Algorithm. Found. Robot. (Zeist, Amsterdam 2004)
- 5.48 K.E. Bekris, L.E. Kavraki: Greedy but safe replanning under differential constraints, IEEE Int. Conf. Robot. Autom. (2007)
- 5.49 C. O'Dunlaing, C.K. Yap: A retraction method for planning the motion of a disc, J. Algorithms **6**, 104–111 (1982)
- 5.50 D. Leven, M. Sharir: Planning a purely translational motion for a convex object in two-dimensional space using generalized Voronoi diagrams, Discrete Comput. Geom. **2**, 9–31 (1987)
- 5.51 M. Sharir: Algorithmic motion planning. In: *Handbook of Discrete and Computational Geometry*, 2nd edn., ed. by J. E. Goodman, J. O'Rourke (Chapman Hall/CRC Press, New York 2004) pp. 1037–1064
- 5.52 N.J. Nilsson: A mobile automaton: An application of artificial intelligence techniques, 1st Int. Conf. Artif. Intell. (1969) pp. 509–520
- 5.53 J. O'Rourke: Visibility. In: *Handbook of Discrete and Computational Geometry*, 2nd edn., ed. by J. E. Goodman, J. O'Rourke (Chapman Hall/CRC Press, New York 2004) pp. 643–663
- 5.54 B. Chazelle: Approximation and decomposition of shapes. In: *Algorithmic and Geometric Aspects of Robotics*, ed. by J.T. Schwartz, C.K. Yap (Lawrence Erlbaum, Hillsdale 1987) pp. 145–185
- 5.55 M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf: *Computational Geometry: Algorithms and Applications*, 2nd edn. (Springer, Berlin 2000)
- 5.56 J.M. Keil: Polygon decomposition. In: *Handbook on Computational Geometry*, ed. by J.R. Sack, J. Urrutia (Elsevier, New York 2000)
- 5.57 J.T. Schwartz, M. Sharir: On the piano movers' problem: I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers, Commun. Pure Appl. Math. **36**, 345–398 (1983)
- 5.58 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robot. Res. **5**(1), 90–98 (1986)
- 5.59 J. Barraquand, J.-C. Latombe: Robot motion planning: A distributed representation approach, Int. J. Robot. Res. **10**(6), 628–649 (1991)
- 5.60 E. Rimon, D.E. Koditschek: Exact robot navigation using artificial potential fields, IEEE Trans. Robot. Autom. **8**(5), 501–518 (1992)
- 5.61 J.P. Laumond: Trajectories for mobile robots with kinematic and environment constraints, Int. Conf. Intell. Auton. Syst. (1986) pp. 346–354
- 5.62 J.P. Laumond, S. Sekhavat, F. Lamiroux: Guidelines in nonholonomic motion planning for mobile robots. In: *Robot Motion Planning and Control*, ed. by J.P. Laumond (Springer, Berlin 1998) pp. 1–53
- 5.63 B.R. Donald, P.G. Xavier, J. Canny, J. Reif: Kinodynamic planning, J. ACM **40**, 1048–1066 (1993)
- 5.64 C. O'Dunlaing: Motion planning with inertial constraints, Algorithmica **2**(4), 431–475 (1987)

- 5.65 J. Canny, A. Rege, J. Reif: An exact algorithm for kinodynamic planning in the plane, *Discrete Comput. Geom.* **6**, 461–484 (1991)
- 5.66 J. Go, T. Vu, J.J. Kuffner: Autonomous behaviors for interactive vehicle animations, *SIGGRAPH Symp. Comput. Animat.* (2004)
- 5.67 M. Pivtoraiko, A. Kelly: Generating near minimal spanning control sets for constrained motion planning in discrete state spaces, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* (2005)
- 5.68 J. Hollerbach: Dynamic scaling of manipulator trajectories, *Tech. Rep. 700* (MIT A.I. Lab Memo, 1983)
- 5.69 K.G. Shin, N.D. McKay: Minimum-time control of robot manipulators with geometric path constraints, *IEEE Trans. Autom. Contr.* **30**(6), 531–541 (1985)
- 5.70 K.G. Shin, N.D. McKay: A dynamic programming approach to trajectory planning of robotic manipulators, *IEEE Trans. Autom. Contr.* **31**(6), 491–500 (1986)
- 5.71 S. Sastry: *Nonlinear Systems: Analysis, Stability, and Control* (Springer, Berlin 1999)
- 5.72 D.J. Balkcom, M.T. Mason: Time optimal trajectories for bounded velocity differential drive vehicles, *Int. J. Robot. Res.* **21**(3), 199–217 (2002)
- 5.73 P. Souères, J.-D. Boissonnat: Optimal trajectories for nonholonomic mobile robots. In: *Robot Motion Planning and Control*, ed. by J.P. Laumond (Springer, Berlin 1998) pp. 93–169
- 5.74 P. Svestka, M.H. Overmars: Coordinated motion planning for multiple car-like robots using probabilistic roadmaps, *IEEE Int. Conf. Robot. Autom.* (1995) pp. 1631–1636
- 5.75 S. Sekhavat, P. Svestka, J.-P. Laumond, M.H. Overmars: Multilevel path planning for nonholonomic robots using semiholonomic subsystems, *Int. J. Robot. Res.* **17**, 840–857 (1998)
- 5.76 P. Ferbach: A method of progressive constraints for nonholonomic motion planning, *IEEE Int. Conf. Robot. Autom.* (1996) pp. 2949–2955
- 5.77 S. Pancanti, L. Pallottino, D. Salvadorini, A. Bicchi: Motion planning through symbols and lattices, *IEEE Int. Conf. Robot. Autom.* (2004) pp. 3914–3919
- 5.78 J. Barraquand, J.-C. Latombe: Nonholonomic multi-body mobile robots: controllability and motion planning in the presence of obstacles, *Algorithmica* **10**, 121–155 (1993)
- 5.79 S.M. LaValle, J.J. Kuffner: Randomized kinodynamic planning, *IEEE Int. Conf. Robot. Autom.* (1999) pp. 473–479
- 5.80 A. M. Ladd, L. E. Kavraki: Motion planning in the presence of drift underactuation and discrete system changes. In: *Robotics: Science and Systems I* ed. by (MIT Press, Boston 2005) pp. 233–241
- 5.81 J.-P. Merlet: *Parallel Robots* (Kluwer, Boston 2000)
- 5.82 D. Cox, J. Little, D. O'Shea: *Ideals, Varieties, and Algorithms* (Springer, Berlin 1992)
- 5.83 R.J. Milgram, J.C. Trinkle: The geometry of configuration spaces for closed chains in two and three dimensions, *Homol. Homot. Appl.* **6**(1), 237–267 (2004)
- 5.84 J. Yakey, S.M. LaValle, L.E. Kavraki: Randomized path planning for linkages with closed kinematic chains, *IEEE Trans. Robot. Autom.* **17**(6), 951–958 (2001)
- 5.85 L. Han, N.M. Amato: A kinematics-based probabilistic roadmap method for closed chain systems. In: *Algorithmic and Computational Robotics: New Directions*, ed. by B.R. Donald, K.M. Lynch, D. Rus (A.K. Peters, Wellesley 2001) pp. 233–246
- 5.86 J. Cortés: Motion Planning Algorithms for General Closed-Chain Mechanisms. Ph.D. Thesis (Institut National Polytechnique de Toulouse, Toulouse 2003)
- 5.87 R. Alami, J.-P. Laumond, T. Siméon: Two manipulation planning algorithms. In: *Algorithms for Robotic Motion and Manipulation*, ed. by J.P. Laumond, M. Overmars (A.K. Peters, Wellesley 1997)
- 5.88 L.E. Kavraki, M. Kolountzakis: Partitioning a planar assembly into two connected parts is NP-complete, *Inform. Process. Lett.* **55**(3), 159–165 (1995)
- 5.89 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)
- 5.90 K. Sutner, W. Maass: Motion planning among time dependent obstacles, *Acta Informatica* **26**, 93–122 (1988)
- 5.91 J.H. Reif, M. Sharir: Motion planning in the presence of moving obstacles, *J. ACM* **41**, 764–790 (1994)
- 5.92 M.A. Erdmann, T. Lozano-Pérez: On multiple moving objects, *Algorithmica* **2**, 477–521 (1987)
- 5.93 J. van den Berg, M. Overmars: Prioritized motion planning for multiple robots, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* (2005) pp. 2217–2222
- 5.94 T. Siméon, S. Leroy, J.-P. Laumond: Path coordination for multiple mobile robots: A resolution complete algorithm, *IEEE Trans. Robot. Autom.* **18**(1), 42–49 (2002)
- 5.95 R. Ghrist, J.M. O'Kane, S.M. LaValle: Pareto optimal coordination on roadmaps, *Workshop Algorithm. Found. Robot.* (2004) pp. 185–200
- 5.96 S.M. LaValle, S.A. Hutchinson: Optimal motion planning for multiple robots having independent goals, *IEEE Trans. Robot. Autom.* **14**(6), 912–925 (1998)
- 5.97 W.M. Boothby: *An Introduction to Differentiable Manifolds and Riemannian Geometry*, 2nd edn. (Academic, New York 2003)
- 5.98 A. Hatcher: *Algebraic Topology* (Cambridge Univ Press, Cambridge 2002)
- 5.99 G.S. Chirikjian, A.B. Kyatkin: *Engineering Applications of Noncommutative Harmonic Analysis* (CRC, Boca Raton 2001)
- 5.100 J. Arvo: Fast random rotation matrices. In: *Graphics Gems III*, ed. by D. Kirk (Academic, New York 1992) pp. 117–120
- 5.101 H. Niederreiter: *Random Number Generation and Quasi-Monte-Carlo Methods* (Society for Industrial and Applied Mathematics, Philadelphia 1992)

- 5.102 S. Basu, R. Pollack, M.-F. Roy: *Algorithms in Real Algebraic Geometry* (Springer, Berlin 2003)
- 5.103 B. Mishra: Computational real algebraic geometry. In: *Handbook of Discrete and Computational Geometry*, ed. by J.E. Goodman, J. O'Rourke (CRC, New York 1997) pp. 537–556
- 5.104 J.T. Schwartz, M. Sharir: On the piano movers' problem: II. General techniques for computing topological properties of algebraic manifolds, *Commun. Pure Appl. Math.* **36**, 345–398 (1983)
- 5.105 B.R. Donald: A search algorithm for motion planning with six degrees of freedom, *Artif. Intell. J.* **31**, 295–353 (1987)
- 5.106 D.S. Arnon: Geometric reasoning with logic and algebra, *Artif. Intell. J.* **37**(1–3), 37–60 (1988)
- 5.107 G.E. Collins: Quantifier elimination by cylindrical algebraic decomposition—twenty years of progress. In: *Quantifier Elimination and Cylindrical Algebraic Decomposition*, ed. by B.F. Caviness, J.R. Johnson (Springer, Berlin 1998) pp. 8–23

Motion Control

6. Motion Control

Wankyun Chung, Li-Chen Fu, Su-Hau Hsu[†]

This Chapter will focus on the motion control of robotic rigid manipulators. In other words, this Chapter does not treat the motion control of mobile robots, flexible manipulators, and manipulators with elastic joints. The main challenge in the motion control problem of rigid manipulators is the complexity of their dynamics and uncertainties. The former results from nonlinearity and coupling in the robot manipulators. The latter is twofold: structured and unstructured. Structured uncertainty means imprecise knowledge of the dynamic parameters and will be touched upon in this Chapter, whereas unstructured uncertainty results from joint and link flexibility, actuator dynamics, friction, sensor noise, and unknown environment dynamics, and will be treated in other Chapters.

In this Chapter, we begin with an introduction to motion control of robot manipulators from a fundamental viewpoint, followed by a survey and brief review of the relevant advanced materials. Specifically, the dynamic model and useful properties of robot manipulators are recalled in Sect. 6.1. The joint and operational space control approaches, two different viewpoints on control of robot manipulators, are compared in Sect. 6.2. Independent joint control and proportional–integral–derivative (PID) control, widely adopted in the field of industrial robots, are presented in Sections 6.3 and 6.4, respectively. Tracking control, based on feedback linearization, is introduced in Sect. 6.5. The computed–torque control and its variants are described in Sect. 6.6. Adaptive control is introduced in Sect. 6.7 to solve the problem of structural uncertainty, whereas the optimality and robustness issues are covered in Sect. 6.8. Since most controllers of robot manipulators are implemented by using microprocessors, the issues of digital implementation are discussed in Sect. 6.9. Finally, learning control, one popular approach to intelligent control, is illustrated in Sect. 6.10.

6.1	Introduction to Motion Control	134
6.1.1	Dynamical Model	134
6.1.2	Control Tasks	135
6.1.3	Summary	135
6.2	Joint Space Versus Operational Space Control	135
6.2.1	Joint Space Control	135
6.2.2	Operational Space Control	136
6.3	Independent-Joint Control	137
6.3.1	Controller Design Based on the Single-Joint Model	137
6.3.2	Controller Design Based on the Multijoint Model	138
6.3.3	Summary	139
6.4	PID Control	139
6.4.1	PD Control for Regulation	139
6.4.2	PID Control for Regulation	140
6.4.3	PID Gain Tuning	140
6.5	Tracking Control	141
6.5.1	Inverse Dynamics Control	141
6.5.2	Feedback Linearization	142
6.5.3	Passivity-Based Control	143
6.5.4	Summary	143
6.6	Computed-Torque Control	143
6.6.1	Computed-Torque Control	143
6.6.2	Computed-Torque-Like Control	144
6.6.3	Summary	146
6.7	Adaptive Control	147
6.7.1	Adaptive Computed-Torque Control	147
6.7.2	Adaptive Inertia-Related Control	147
6.7.3	Adaptive Control Based on Passivity	148
6.7.4	Adaptive Control with Desired Compensation	149
6.7.5	Summary	149
6.8	Optimal and Robust Control	150
6.8.1	Quadratic Optimal Control	150
6.8.2	Nonlinear \mathcal{H}_∞ Control	151
6.8.3	Passivity-Based Design of Nonlinear \mathcal{H}_∞ Control	151

6.8.4 A Solution to Inverse Nonlinear \mathcal{H}_∞ Control ...	152	6.10 Learning Control.....	155
6.9 Digital Implementation.....	153	6.10.1 Pure P-type Learning Control.....	156
6.9.1 Reference Trajectory Generation.....	153	6.10.2 P-Type Learning Control with a Forgetting Factor.....	156
6.9.2 Z-Transform for Coding.....	154	6.10.3 Summary	157
		References	157

6.1 Introduction to Motion Control

The dynamical model of robot manipulators will be recalled in this section. Furthermore, important properties of this dynamical model, which is useful in controller design, will then be addressed. Finally, different control tasks of the robot manipulators will be defined.

6.1.1 Dynamical Model

For motion control, the dynamical model of rigid robot manipulators is conveniently described by Lagrange dynamics. Let the robot manipulator have n links and let the $(n \times 1)$ -vector \mathbf{q} of joint variables be $\mathbf{q} = [q_1, \dots, q_n]^\top$. The dynamic model of the robot manipulator is then described by Lagrange's equation [6.1–6]:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}) = \boldsymbol{\tau}, \quad (6.1)$$

where $\mathbf{H}(\mathbf{q})$ is the $(n \times n)$ inertia matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ is the $(n \times 1)$ -vector of Coriolis and centrifugal forces, $\boldsymbol{\tau}_g(\mathbf{q})$ is the $(n \times 1)$ -vector of gravity force, and $\boldsymbol{\tau}$ is the $(n \times 1)$ -vector of joint control inputs to be designed. Friction and disturbance input have been neglected here.

Remark.

- Other contributions to the dynamic description of the robot manipulators may include the dynamics of the actuators, joint and link flexibility, friction, noise, and disturbances. Without loss of generality, the case of the rigid robot manipulators is stressed here.

The control schemes that we will introduce in this Chapter are based on some important properties of the dynamical model of robot manipulators. Before giving a detailed introduction to these different schemes, let us first give a list of those properties.

Property 6.1

The inertia matrix is a symmetric positive-definite matrix, which can be expressed

$$\lambda_h \mathbf{I}_n \leq \mathbf{H}(\mathbf{q}) \leq \lambda_H \mathbf{I}_n, \quad (6.2)$$

where λ_h and λ_H denote positive constants.

Property 6.2

The matrix $\mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{\mathbf{H}}(\mathbf{q}) - 2\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is skew-symmetric for a particular choice of $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ (which is always possible), i. e.,

$$\mathbf{z}^\top \mathbf{N}(\mathbf{q}, \dot{\mathbf{q}}) \mathbf{z} = 0 \quad (6.3)$$

for any $(n \times 1)$ -vector \mathbf{z} .

Property 6.3

The $(n \times n)$ -matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ satisfies

$$\|\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\| \leq c_o \|\dot{\mathbf{q}}\| \quad (6.4)$$

for some bounded constant c_o .

Property 6.4

The gravity force/torque vector satisfies

$$\|\boldsymbol{\tau}_g(\mathbf{q})\| \leq g_o \quad (6.5)$$

for some bounded constant g_o .

Property 6.5

The equation of motion is linear in the inertia parameters. In other words, there is a $(r \times 1)$ constant vector \mathbf{a} and an $(n \times r)$ regressor matrix $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ such that

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}) = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\mathbf{a}. \quad (6.6)$$

The vector \mathbf{a} is comprised of link masses, moments of inertia, and the link, in various combinations.

Property 6.6

The mapping $\boldsymbol{\tau} \rightarrow \dot{\mathbf{q}}$ is passive; i.e., there exists $\alpha \geq 0$ such that

$$\int_0^t \dot{\mathbf{q}}^\top(\beta) \boldsymbol{\tau}(\beta) d\beta \geq -\alpha, \quad \forall t < \infty. \quad (6.7)$$

Remarks.

- Properties 6.3 and 6.4 are very useful since they allow us to establish upper bounds on the nonlinear terms in the dynamical model. As we will see further, several control schemes require knowledge of such upper bounds.
- In Property 6.5, the parameter vector \mathbf{a} is comprised of several variables in various combinations. The dimensionality of the parameter space is not unique, and the search over the parameter space is an important problem.
- In this section, we assume that *the robot manipulator is fully actuated* and this indicates that there is an independent control input for each degree of freedom (DOF). In contrast, the robot manipulators with joint or link flexibility are no longer fully actuated and the control problems are more difficult in general.

6.1.2 Control Tasks

It is instructive for comparative purposes to classify control objectives into the following two classes:

- *Trajectory tracking* is aimed at following a time-varying joint reference trajectory specified within the manipulator workspace. In general, this desired trajectory is assumed to comply with the actuators' capacity. In other words, the joint velocity and acceleration associated with the desired trajectory should

not violate, respectively, the velocity and acceleration limit of the manipulator. In practice, the capacity of actuators is set by torque limits, which result in bounds on the acceleration that are complex and state dependent.

- *Regulation* sometimes is also called point-to-point control. A fixed configuration in the joint space is specified; the objective is to bring to and keep the joint variable at the desired position in spite of torque disturbances and independently of the initial conditions. The behavior of transients and overshooting, are in general, not guaranteed.

The selection of the controller may depend on the type of task to be performed. For example, tasks only requiring the manipulator to move from one position to another without requiring significant precision during the motion between these two points can be solved by regulators, whereas such as welding, painting, and so on, require tracking controllers.

Remarks.

- The regulation problem may be seen as a special case of the tracking problem (for which the desired joint velocity and acceleration are zero).
- The task specification above is given in the joint space and results in joint space control, which is the main content of this Chapter. Sometimes, the task specification of the robot manipulators in terms of the desired trajectory of the end-effector (e.g., control with eye-in-hand) is carried out in the task space and gives rise to the operational space control, which will be introduced in Sect. 6.2.

6.1.3 Summary

In this section, we introduced the dynamical model of the robot manipulators and important properties of this dynamical model. Finally, we defined different control tasks of the robot manipulators.

6.2 Joint Space Versus Operational Space Control

In a motion control problem, the manipulator moves to a position to pick up an object, transports that object to another location, and deposits it. Such a task is an integral part of any higher-level manipulation tasks such as painting or spot-welding.

Tasks are usually specified in the task space in terms of a desired trajectory of the end-effector, while control actions are performed in the joint space to achieve the desired goals. This fact naturally leads to two kinds of general control methods, namely *joint space con-*

trol and operational space control (task space control) schemes.

6.2.1 Joint Space Control

The main goal of the joint space control is to design a feedback controller such that the joint coordinates $\mathbf{q}(t) \in \mathbb{R}^n$ track the desired motion $\mathbf{q}_d(t)$ as closely as possible. To this end, consider the equations of motion (6.1) of an n -DOF manipulator expressed in the joint space [6.2, 4]. In this case, the control of robot manipulators is naturally achieved in the joint space, since the control inputs are the joint torques. Nevertheless, the user specifies a motion in terms of end-effector coordinates, and thus it is necessary to understand the following strategy.

Figure 6.1 shows the basic outline of the joint space control methods. Firstly, the desired motion, which is described in terms of end-effector coordinates, is converted to a corresponding joint trajectory using the inverse kinematics of the manipulator. Then the feedback controller determines the joint torque necessary to move the manipulator along the desired trajectory specified in joint coordinates starting from measurements of the current joint states [6.1, 4, 7, 8].

Since it is always assumed that the desired task is given in terms of the time sequence of the joint motion, joint space control schemes are quite adequate in situations where manipulator tasks can be accurately preplanned and little or no online trajectory adjustments are necessary [6.1, 4, 7, 9]. Typically, inverse kinematics is performed for some intermediate task points, and the joint trajectory is interpolated using the intermediate joint solutions. Although the command trajectory consists of straight-line motions in end-effector coordinates between interpolation points, the resulting joint motion consists of curvilinear segments that match

the desired end-effector trajectory at the interpolation points.

In fact, the joint space control includes simple PD control, PID control, inverse dynamic control, Lyapunov-based control, and passivity-based control, as explained in the following sections.

6.2.2 Operational Space Control

In more complicated and less certain environments, end-effector motion may be subject to online modifications in order to accommodate unexpected events or to respond to sensor inputs. There are a variety of tasks in manufacturing where these type of control problem arise. In particular, it is essential when controlling the interaction between the manipulator and environment is of concern.

Since the desired task is often specified in the operational space and requires precise control of the end-effector motion, joint space control schemes are not suitable in these situations. This motivated a different approach, which can develop control schemes directly based on the dynamics expressed in the operational space [6.10, 11].

Let us suppose that the Jacobian matrix, denoted by $\mathbf{J}(\mathbf{q}) \in \mathbb{R}^{n \times n}$, transforms the joint velocity ($\dot{\mathbf{q}} \in \mathbb{R}^n$) to the task velocity ($\dot{\mathbf{x}} \in \mathbb{R}^n$) according to

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \quad (6.8)$$

Furthermore, assume that it is invertible. Then, the operational space dynamics is expressed as follows:

$$\mathbf{f}_c = \mathbf{A}(\mathbf{q})\ddot{\mathbf{x}} + \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}), \quad (6.9)$$

where $\mathbf{f}_c \in \mathbb{R}^n$ denotes the command forces in the operational space; the *pseudo-inertia matrix* is defined by

$$\mathbf{A}(\mathbf{q}) = \mathbf{J}^{-\top}(\mathbf{q})\mathbf{H}(\mathbf{q})\mathbf{J}^{-1}(\mathbf{q}), \quad (6.10)$$

and $\mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}})$ and $\boldsymbol{\eta}(\mathbf{q})$ are given by

$$\begin{aligned} \mathbf{\Gamma}(\mathbf{q}, \dot{\mathbf{q}}) &= \mathbf{J}^{-\top}(\mathbf{q})\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{J}^{-1}(\mathbf{q}) - \mathbf{A}(\mathbf{q})\mathbf{J}(\mathbf{q})\mathbf{J}^{-1}(\mathbf{q}), \\ \boldsymbol{\eta}(\mathbf{q}) &= \mathbf{J}^{-\top}(\mathbf{q})\boldsymbol{\tau}_g(\mathbf{q}). \end{aligned}$$

The task space variables are usually reconstructed from the joint space variables, via the kinematic mappings. In fact, it is quite rare to have sensors to directly measure end-effector positions and velocities. Also, it is worth remarking that an analytical Jacobian is utilized since the control schemes operate directly on task space quantities, i. e., the end-effector position and orientation.

The main goal of the operational space control is to design a feedback controller that allows execution of an end-effector motion $\mathbf{x}(t) \in \mathbb{R}^n$ that tracks the de-

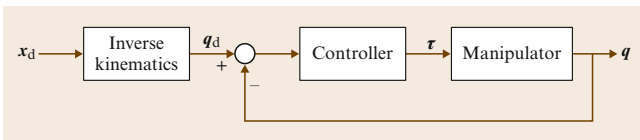


Fig. 6.1 Generic concept of joint space control

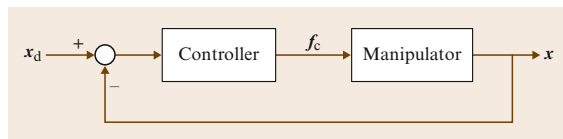


Fig. 6.2 Basic concept of operational space control

sired end-effector motion $\mathbf{x}_d(t)$ as closely as possible. To this end, consider the equations of motion (6.9) of the manipulator expressed in the operational space. For this case, Fig. 6.2 shows a schematic diagram of the operational space control methods. There are several advantages to such an approach because operational

space controllers employ a feedback loop that directly minimizes task errors. Inverse kinematics need not be calculated explicitly, since the control algorithm embeds the velocity-level forward kinematics (6.8), as shown in the figure. Now, motion between points can be a straight-line segment in the task space.

6.3 Independent-Joint Control

By independent-joint control (i.e., decentralized control), we mean that the control inputs of each joint only depends on the measurement of the corresponding joint displacement and velocity. Due to its simple structure, this kind of control schemes offers many advantages. For example, by using independent-joint control, communication among different joints is saved. Moreover, since the computational load of controllers may be reduced, only low-cost hardware is required in actual implementations. Finally, independent-joint control has the feature of scalability, since the controllers on all the joints have the same formulation. In this section, two kinds of design of independent-joint control will be introduced: one focused on the dynamical model of each joint (i.e., based on the single-joint model) and the other based on the analysis of the overall dynamical model (i.e., the multi-joint model) of robot manipulators.

6.3.1 Controller Design Based on the Single-Joint Model

The simplest independent-joint control strategy is to control each joint axis as a single-input single-output (SISO) system. Coupling effects among joints due to varying configuration during motion are treated as disturbance inputs. Without loss of generality, the actuator is taken as a rotary electric direct-current (DC) motor. Hence, the block diagram of the control scheme of joint i can be represented in the domain of the complex variables as shown in Fig. 6.3. In this scheme, θ is the angular variable of the motor, J is the effective inertia viewed from the motor side, R_a is the armature resistance (auto-inductance being neglected), and k_t and k_v are, respectively, the torque and motor constants. Furthermore, G_v denotes the voltage gain of the power amplifier so that the reference input is the input voltage V_c of the amplifier instead of the armature voltage V_a . It has also been assumed that $F_m \ll k_v k_t / R_a$, i.e., the mechanical (viscous) friction coefficient has

been neglected with respect to the electrical coefficient. Now the input–output transfer function of the motor can be written as

$$M(s) = \frac{k_m}{s(1 + sT_m)}, \quad (6.11)$$

where $k_m = G_v/k_v$ and $T_m = R_a J/k_v k_t$ are, respectively, the voltage-to-velocity gain and the time constant of the motor.

To guide selection of the control structure, start by noticing that an effective rejection of the disturbance d on the output θ is ensured by

1. a large value of the amplifier before the point of intervention of the disturbance;
2. the presence of an integral action in the controller so as to cancel the effect of the gravitational component on the output at the steady state (i.e., constant θ).

In this case, as shown in Fig. 6.4, the types of control action with position and velocity feedback are characterized by [6.4]

$$G_p(s) = K_P, \quad G_v(s) = K_V \frac{1 + sT_V}{s}, \quad (6.12)$$

where $G_p(s)$ and $G_v(s)$ correspond to the position and velocity control actions, respectively. It is worth noting that the inner control action $G_v(s)$ is in a form of propositional–integral (PI) control to yield zero error in the steady state under a constant disturbance d . Furthermore, k_{TP} and k_{TV} are both transducer constants, and the amplifier gain K_V has been embedded in the gain

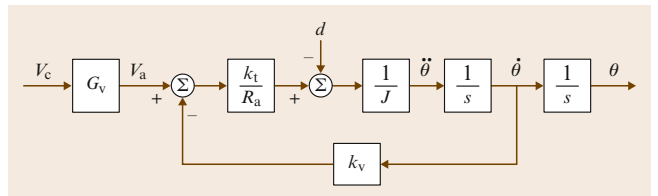


Fig. 6.3 Block scheme of a joint-driven system (after [6.4])

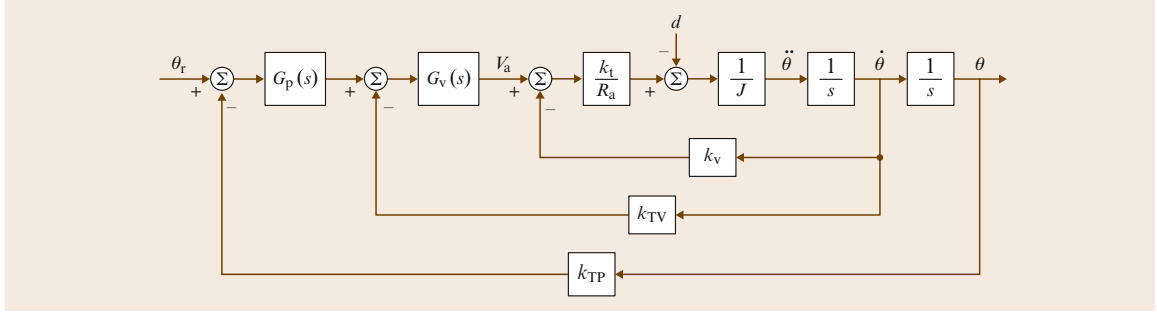


Fig. 6.4 Block scheme of position and velocity feedback (after [6.4])

of the inner controller. From the scheme of Fig. 6.4, the transfer function of the forward path is

$$P(s) = \frac{k_m K_P K_V (1 + sT_V)}{s(1 + sT_m)}, \quad (6.13)$$

while that of the return path is

$$H(s) = k_{TP} \left(1 + s \frac{k_{TV}}{K_P k_{TP}} \right). \quad (6.14)$$

The zero of the controller at $s = -1/T_V$ can be chosen so as to cancel the effects of the real pole of the motor at $s = -1/T_m$. Then, by setting $T_V = T_m$, the poles of the closed-loop system move on the root locus as a function of the loop gain, $k_m K_V k_{TV}$. By increasing the position feedback gain K_P , it is possible to confine the closed-loop poles to a region of the complex plane with large absolute real part. Then, the actual location can be established by a suitable choice of K_V .

The closed-loop input–output transfer function is

$$\frac{\Theta(s)}{\Theta_r(s)} = \frac{\frac{1}{k_{TP}}}{1 + \frac{sk_{TP}}{K_P k_{TP}} + \frac{s^2}{k_m K_P k_{TP} K_V}}, \quad (6.15)$$

which can be compared with the typical transfer function of a second-order system

$$W(s) = \frac{\frac{1}{k_{TP}}}{1 + \frac{2\zeta s}{\omega_n} + \frac{s^2}{\omega_n^2}}. \quad (6.16)$$

It can be recognized that, with a suitable choice of the gains, it is possible to obtain any value of natural frequency ω_n and damping ratio ζ . Hence, if ω_n and ζ are given as design specifications, the following relations can be found

$$K_V k_{TV} = \frac{2\zeta\omega_n}{k_m} \quad \text{and} \quad K_P k_{TP} K_V = \frac{\omega_n^2}{k_m}. \quad (6.17)$$

For given transducer constants k_{TP} and k_{TV} , K_V and K_P will be chosen to satisfy the two equations above, respectively. On the other hand, the closed-loop disturbance/output function is

$$\frac{\Theta(s)}{D(s)} = \frac{\frac{sR_a}{k_t K_P K_{TP} K_V (1 + sT_m)}}{1 + \frac{sk_{TV}}{K_P k_{TP}} + \frac{s^2}{k_m K_P k_{TP} K_V}}, \quad (6.18)$$

which shows that the disturbance rejection factor is $X_R(s) = K_P k_{TP} K_V$ and is fixed, provided that K_V and K_P have been chosen via the approach above. Concerning the disturbance dynamics, the zero at the origin introduced by the PI, a real pole at $s = -1/T_m$, and the pair of complex poles with real part $-\zeta\omega_n$ should be kept in mind. In this case, an estimate T_R of the output recovery time needed by the control system to recover from the effect of a disturbance on the joint position can be evaluated by analyzing models of the transfer function above. Such an estimate can reasonably be expressed as $T_R = \max\{T_m, 1/\zeta\omega\}$.

6.3.2 Controller Design Based on the Multijoint Model

In recent years, independent-joint control schemes based on the complete dynamic model of the robot manipulators (i.e., a multijoint model) have been proposed. For example, following the approach of computed-torque-like control, [6.12] dealt with the regulation task for horizontal motions, and [6.13] and [6.14] handled the tracking task for arbitrary smooth trajectories. Since the overall dynamic model is considered, the effects of coupling among joints are handled. These schemes will be introduced in detail in Sect. 6.6.

6.3.3 Summary

In this section, we have presented two independent-joint control schemes: one is based on the single-joint model and the other is based on the multijoint model. The former focuses on the dynamics of a single joint and regards the interaction among joints as a disturbance. This control scheme is simple but may not be suitable for high-speed tracking. Hence, we introduce the latter, which considers the overall dynamical model of robot manipulators such that the interaction among joints can be handled.

6.4 PID Control

Traditionally, control design in robot manipulators can be understood as the simple fact of tuning of a PD or PID compensator at the level of each motor driving the manipulator joints [6.1]. Fundamentally, a PD controller is a position and velocity feedback that has good closed-loop properties when applied to a double integrator system.

The PID control has a long history since Ziegler and Nichols' PID tuning rules were published in 1942 [6.15]. Actually, the strong point of PID control lies in its *simplicity* and clear physical meaning. Simple control is preferable to complex control, at least in industry, if the performance enhancement obtained by using complex control is not significant enough. The physical meanings of PID control [6.16] are as follows:

- P-control means the present effort making a present state into desired state;
- I-control means the accumulated effort using the experience information of previous states;
- D-control means the predictive effort reflecting the information about trends in future states.

6.4.1 PD Control for Regulation

A simple design method for manipulator control is to utilize a linear control scheme based on the linearization of the system about an operating point. An example of this method is a PD control with a gravity compensation scheme [6.17, 18]. Gravity compensation acts as a bias correction, compensating only for the amount of forces that create overshooting and an asymmetric transient behavior. Formally, it has the following form

$$\tau = K_P(q_d - q) - K_V\dot{q} + \tau_g(q), \quad (6.19)$$

Further Reading

There are different types of feedback applied in the independent-joint control based on the single-joint model (such as pure position feedback or position, velocity, and acceleration feedback). A complete discussion is given in [6.4]. When the joint control servos are required to track reference trajectories with high speeds and accelerations, the tracking capabilities of the above schemes are inevitably degraded. A possible remedy is to adopt decentralized feed-forward compensation to reduce tracking errors [6.4, 5].

where K_P and $K_V \in R^{n \times n}$ are positive-definite gain matrices. This controller is very useful for set-point regulation, i. e., $q_d = \text{constant}$ [6.7, 18]. When this controller is applied to (6.1), the closed-loop equation becomes

$$H(q)\ddot{q} + C(q, \dot{q})\dot{q} + K_V\dot{q} - K_P e_q = 0, \quad (6.20)$$

where $e_q = q_d - q$, and the equilibrium point is $y = [e_q^\top, \dot{q}^\top]^\top = 0$. Now, the stability achieved by PD control with gravity compensation can be analyzed according to the closed-loop dynamic (6.20). Consider the positive-definite function

$$V = \frac{1}{2}\dot{q}^\top H(q)\dot{q} + \frac{1}{2}e_q^\top K_V e_q.$$

Then, the derivative of function becomes negative semidefinite for any value of \dot{q} by using Property 6.2 in Sect. 6.1, i. e.,

$$\dot{V} = -\dot{q}^\top K_V \dot{q} \leq -\lambda_{\min}(K_V)\|\dot{q}\|^2, \quad (6.21)$$

where $\lambda_{\min}(K_V)$ means the smallest eigenvalue of K_V . By invoking the Lyapunov stability theory and LaSalle's theorem [6.1], it can be shown that the regulation error will converge asymptotically to zero, while their high-order derivatives remain bounded. This controller requires knowledge of the gravity components (structure and parameters), though it is simple.

Now, consider simple PD control without gravity compensation

$$\tau = K_P(q_d - q) - K_V\dot{q}, \quad (6.22)$$

then the closed-loop dynamic equation becomes

$$H(q)\ddot{q} + C(q, \dot{q})\dot{q} + \tau_g(q) + K_V\dot{q} - K_P e_q = 0. \quad (6.23)$$

Consider the positive definite function

$$V = \frac{1}{2} \dot{\mathbf{q}}^\top \mathbf{H}(\mathbf{q}) \dot{\mathbf{q}} + \frac{1}{2} \mathbf{e}_q^\top \mathbf{K}_V \mathbf{e}_q + U(\mathbf{q}) + U_0,$$

where $U(\mathbf{q})$ denotes the potential energy with the relation of $\partial U(\mathbf{q})/\partial \mathbf{q} = \boldsymbol{\tau}_g(\mathbf{q})$ and U_0 is a suitable constant. Taking time derivative of V along the closed-loop dynamics (6.23) gives the same result (6.21) with previous one using gravity compensation. In this case, the control system must be stable in the sense of Lyapunov, but it can not conclude that the regulation error will converge to zero by LaSalle's theorem [6.1]. Actually, the system precision (the size of the regulation error vector) will depend on the size of the gain matrix \mathbf{K}_P in the following form:

$$\|\mathbf{e}_q\| \leq \|\mathbf{K}_P^{-1}\| g_0, \quad (6.24)$$

where g_0 is that in Property 6.4 in Sect. 6.1. Hence, the regulation error can be arbitrarily reduced by increasing \mathbf{K}_P ; nevertheless, measurement noise and other unmodeled dynamics, such as actuator friction, will limit the use of high gains in practice.

6.4.2 PID Control for Regulation

An integral action may be added to the previous PD control in order to deal with gravity forces, which to some extent can be considered as a constant disturbance (from the local point of view). The PID regulation controller can be written in the following general form:

$$\boldsymbol{\tau} = \mathbf{K}_P(\mathbf{q}_d - \mathbf{q}) + \mathbf{K}_I \int f(\mathbf{q}_d - \mathbf{q}) dt - \mathbf{K}_V \dot{\mathbf{q}},$$

where $\mathbf{K}_I \in \mathbb{R}^{n \times n}$ is a positive-definite gain matrix, and

- if $f(\mathbf{q}_d - \mathbf{q}) = \mathbf{q}_d - \mathbf{q}$, we have PID control,
- if $\mathbf{K}_I \int (-\dot{\mathbf{q}}) dt$ is added, we have PI²D control,
- if $f(\cdot) = \tanh(\cdot)$, we have PD + nonlinear integral control.

Global asymptotic stability (GAS) by PID control was proved in [6.12] for robotic motion control system including external disturbances, such as Coulomb friction. Also, Tomei proved the GAS of PD control in [6.19] by using an adaptation for gravity term. On the other hand, Ortega et al. showed in [6.20] that PI²D control could yield semiglobal asymptotic stability (SGAS) in the presence of gravity and bounded external disturbances. Also, Angeli proved in [6.21] that PD control could achieve the input-output-to-state stability (IOSS) for robotic systems. Also, Ramirez et al. proved the SGAS (with some conditions) for PID gains in [6.22].

Also, Kelly proved in [6.23] that PD plus nonlinear integral control could achieve GAS under gravity.

Actually, a large integral action in PID control can cause instability of the motion control system. In order to avoid this, the integral gain should be bounded by an upper limit of the following form [6.1]:

$$\frac{k_P k_V}{\lambda_H^2} > k_I,$$

where λ_H is that in Property 6.1 in Sect. 6.1, $\mathbf{K}_P = k_P \mathbf{I}$, $\mathbf{K}_I = k_I \mathbf{I}$, and $\mathbf{K}_V = k_V \mathbf{I}$. This relation gives the guidelines for gain selection implicitly. Also, PID control has generated a great variety of PID control plus something, e.g., PID plus friction compensator, PID plus gravity compensator, PID plus disturbance observer.

6.4.3 PID Gain Tuning

The PID control can be utilized for trajectory tracking as well as set-point regulation. True tracking control will be treated after Sect. 6.5. In this section, the simple but useful PID gain tuning method will be introduced for practical use. The general PID controller can be written in the following general form:

$$\boldsymbol{\tau} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \mathbf{K}_I \int \mathbf{e}_q dt,$$

or, in another form,

$$\boldsymbol{\tau} = \left(\mathbf{K} + \frac{1}{\gamma^2} \mathbf{I} \right) \left(\dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \mathbf{K}_I \int \mathbf{e}_q dt \right). \quad (6.25)$$

In a fundamental stability analysis of tracking control systems, Qu et al. proved in [6.24] that PD control could satisfy uniform ultimate boundedness (UUB). Also, Berghuis et al. proposed output feedback PD control, which satisfies semiglobal uniform ultimate boundedness (SGUUB) in [6.25] under gravity and a bounded disturbance. Recently, Choi et al. suggested inverse optimal PID control [6.26], assuring extended disturbance input-to-state stability (ISS).

Actually, if a PID controller (6.25) is repeatedly applied to the same set point or desired trajectory, then the maximum error will be proportional to the gains in the following form:

$$\max_{0 \leq t \leq t_f} \|\mathbf{e}_q(t)\| \propto \frac{\gamma^2}{\sqrt{2k\gamma^2 + 1}}, \quad (6.26)$$

where t_f denotes the final execution time of a given task and $\mathbf{K} = k\mathbf{I}$. This relation can be utilized to tune the gain of a PID controller and is referred to as the compound

tuning rule [6.16]. The compound tuning rule implicitly includes simple tuning rules as follows:

$$\begin{aligned} \text{Square tuning: } \max \|e_q\| &\propto \gamma^2, \quad \text{for a small } k, \\ \text{Linear tuning: } \max \|e_q\| &\propto \gamma, \quad \text{for a large } k. \end{aligned}$$

For example, suppose we select positive constant diagonal matrices $K_P = k_P I$, $K_I = k_I I$, while satisfying $k_P^2 > 2k_I$. For small k , the maximum error will be reduced by $\frac{1}{4}$ according to the square tuning rule, if we reduce the value γ by $\frac{1}{2}$. For large k , the maximum error will be proportionally reduced as γ according to the linear tuning rule. This means that we can tune the PID controller using only one parameter γ when the other gain parameters are fixed [6.16]. Although these rules

are very useful in tuning the control performance, they can be utilized only for repetitive experiments for the same set point or desired trajectory because the tuning rules consist of proportional relations.

Further Reading

PID-type controllers were designed to solve the regulation control problem. They have the advantage of requiring knowledge of neither the model structure nor the model parameters. Also, the stability achieved by PID-type controllers was presented in this section. A range of books and papers [6.1, 15, 16, 22, 27, 28] are available to the robotics audience, detailing the individual tuning methods used in PID control and their concrete proofs.

6.5 Tracking Control

While independent PID controls are adequate in most set-point regulation problems, there are many tasks that require effective trajectory tracking capabilities such as plasma-welding, laser-cutting or high-speed operations in the presence of obstacles. In this case, employing local schemes requires moving slowly through a number of intermediate set points, thus considerably delaying the completion of the task. Therefore, to improve the trajectory tracking performance, controllers should take account of the manipulator dynamic model via a computed-torque-like technique.

The tracking control problem in the joint or task space consists of following a given time-varying trajectory $q_d(t)$ or $x_d(t)$ and its successive derivatives $\dot{q}_d(t)$ or $\dot{x}_d(t)$ and $\ddot{q}_d(t)$ or $\ddot{x}_d(t)$, which describe the desired velocity and acceleration, respectively. To obtain successful performance, significant effort has been devoted to the development of model-based control strategies [6.1, 2, 7]. Among the control approaches reported in the literature, typical methods include inverse dynamic control, the feedback linearization technique, and the passivity-based control method.

6.5.1 Inverse Dynamics Control

Though the inverse dynamics control has a theoretical background, such as the theory of feedback linearization discussed later, its starting point is mechanical engineering intuition based on cancelling nonlinear terms and decoupling the dynamics of each link. Inverse dynamics control in joint space has the

form

$$\tau = H(q)v + C(q, \dot{q})\dot{q} + \tau_g(q) \quad (6.27)$$

which, applied to (6.1), yields a set of n decoupled linear systems, e.g., $\ddot{q} = v$, where v is an auxiliary control input to be designed. Typical choices for v are

$$v = \ddot{q}_d + K_V(\dot{q}_d - \dot{q}) + K_P(q_d - q) \quad (6.28)$$

or with an integral component

$$v = \ddot{q}_d + K_V(\dot{q}_d - \dot{q}) + K_P(q_d - q) + K_I \int (q_d - q) dt, \quad (6.29)$$

leading to the error dynamics equation

$$\ddot{e}_q + K_V \dot{e}_q + K_P e_q = 0$$

for an auxiliary control input (6.28), and

$$e_q^{(3)} + K_V \ddot{e}_q + K_P \dot{e}_q + K_I e_q = 0$$

if an auxiliary control input (6.29) is used. Both error dynamics are exponentially stable by a suitable choice of the gain matrices K_V and K_P (and K_I).

Alternatively, inverse dynamics control can be described in the operational space. Consider the operational space dynamics (6.9). If the following inverse dynamics control is used in the operational space,

$$f_c = \Lambda(q)(\ddot{x}_d + K_V \dot{e}_x + K_P e_x) + \Gamma(q, \dot{q})\dot{x} + \eta(q),$$

where $e_x = x_d - x$, the resulting error dynamics is

$$\ddot{e}_x + K_V \dot{e}_x + K_P e_x = 0, \quad (6.30)$$

and it is also exponentially stable. One apparent advantage of using this controller is that \mathbf{K}_P and \mathbf{K}_V can be selected with a clear physical meaning in operational space. However, as can be seen in (6.10), $\mathbf{A}(\mathbf{q})$ becomes very large when the robot approaches singular configurations [6.8]. This means that large forces in some direction are needed to move the arm.

6.5.2 Feedback Linearization

This approach generalizes the concept of inverse dynamics of rigid manipulators. The basic idea of feedback linearization is to construct a transformation as a so-called *inner-loop control*, which exactly linearizes the nonlinear system after a suitable state space change of coordinates. One can then design a second stage or *outer-loop control* in the new coordinates to satisfy the traditional control design specifications such as tracking, disturbance rejection, etc. [6.5, 29]. The full power of the feedback linearization scheme for manipulator control becomes apparent if one includes in the dynamic description of the manipulator the transmission dynamics, such as the elasticity resulting from shaft windup, gear elasticity, etc. [6.5].

In recent years, an impressive volume of literature has emerged in the area of differential-geometric methods for nonlinear systems. Most of the results in this area are intended to give abstract coordinate-free descriptions of various geometric properties of nonlinear systems and as such are difficult for the non-mathematician to follow. It is our intention here to give only the basic idea of the feedback linearization scheme and to introduce a simple version of this technique that finds an immediate application to the manipulator control problem. The reader is referred to [6.30] for a comprehensive treatment of the feedback linearization technique using differential-geometric methods.

Let us now develop a simple approach to the determination of linear state-space representations of the manipulator dynamics (6.1) by considering a general sort of output $\xi \in R^p$:

$$\xi = \mathbf{h}(\mathbf{q}) + \mathbf{r}(t), \quad (6.31)$$

where $\mathbf{h}(\mathbf{q})$ is a general predetermined function of the joint coordinate $\mathbf{q} \in R^n$ and $\mathbf{r}(t)$ is a general predetermined time function. The control objective will be to select the joint torque inputs $\boldsymbol{\tau}$ in order to make the output $\xi(t)$ go to zero.

The choice of $\mathbf{h}(\mathbf{q})$ and $\mathbf{r}(t)$ is based on the control purpose. For example, if $\mathbf{h}(\mathbf{q}) = -\mathbf{q}$ and $\mathbf{r}(t) = \mathbf{q}_d(t)$, the desired joint space trajectory we would like the

manipulator to follow, then $\xi(t) = \mathbf{q}_d(t) - \mathbf{q}(t) \equiv \mathbf{e}_q(t)$ is the joint space tracking error. Forcing $\xi(t)$ to zero in this case would cause the joint variables $\mathbf{q}(t)$ to track their desired values $\mathbf{q}_d(t)$, resulting in a manipulator trajectory-following problem. As another example, $\xi(t)$ could represent the operational space tracking error $\xi(t) = \mathbf{x}_d(t) - \mathbf{x}(t) \equiv \mathbf{e}_x(t)$. Then, controlling $\xi(t)$ to zero would result in trajectory following directly in operational space where the desired motion is usually specified.

To determine a linear state-variable model for manipulator controller design, let us simply differentiate the output $\xi(t)$ twice to obtain

$$\dot{\xi} = \frac{\partial \mathbf{h}}{\partial \mathbf{q}} \dot{\mathbf{q}} + \dot{\mathbf{r}} = \mathbf{T} \dot{\mathbf{q}} + \dot{\mathbf{r}}, \quad (6.32)$$

$$\ddot{\xi} = \mathbf{T} \ddot{\mathbf{q}} + \dot{\mathbf{T}} \dot{\mathbf{q}} + \ddot{\mathbf{r}}, \quad (6.33)$$

where we defined a $(p \times n)$ transformation matrix of the form

$$\mathbf{T}(\mathbf{q}) = \frac{\partial \mathbf{h}(\mathbf{q})}{\partial \mathbf{q}} = \begin{pmatrix} \frac{\partial h_1}{\partial q_1} & \frac{\partial h_1}{\partial q_2} & \cdots & \frac{\partial h_1}{\partial q_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial h_p}{\partial q_1} & \frac{\partial h_p}{\partial q_2} & \cdots & \frac{\partial h_p}{\partial q_n} \end{pmatrix}. \quad (6.34)$$

Given the output $\mathbf{h}(\mathbf{q})$, it is straightforward to compute the transformation $\mathbf{T}(\mathbf{q})$ associated with $\mathbf{h}(\mathbf{q})$. In the special case where $\dot{\xi}$ represents the operational space velocity error, then $\mathbf{T}(\mathbf{q})$ denotes the Jacobian matrix $\mathbf{J}(\mathbf{q})$.

According to (6.1),

$$\ddot{\mathbf{q}} = \mathbf{H}^{-1}(\mathbf{q})[\boldsymbol{\tau} - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}})] \quad (6.35)$$

with the nonlinear terms represented by

$$\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \boldsymbol{\tau}_g(\mathbf{q}). \quad (6.36)$$

Then (6.33) yields

$$\ddot{\xi} = \ddot{\mathbf{r}} + \dot{\mathbf{T}} \dot{\mathbf{q}} + \mathbf{T}(\mathbf{q})\mathbf{H}^{-1}(\mathbf{q})[\boldsymbol{\tau} - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}})]. \quad (6.37)$$

Define the *control input* function

$$\mathbf{u} = \ddot{\mathbf{r}} + \dot{\mathbf{T}} \dot{\mathbf{q}} + \mathbf{T}(\mathbf{q})\mathbf{H}^{-1}(\mathbf{q})[\boldsymbol{\tau} - \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}})]. \quad (6.38)$$

Now we may define a state $\mathbf{y}(t) \in R^{2p}$ by $\mathbf{y} = (\xi \quad \dot{\xi})$ and write the manipulator dynamics as

$$\dot{\mathbf{y}} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_p \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \mathbf{y} + \begin{pmatrix} \mathbf{0} \\ \mathbf{I}_p \end{pmatrix} \mathbf{u}. \quad (6.39)$$

This is a linear state-space system of the form

$$\dot{\mathbf{y}} = \mathbf{A}\mathbf{y} + \mathbf{B}\mathbf{u}, \quad (6.40)$$

driven by the control input \mathbf{u} . Due to the special form of \mathbf{A} and \mathbf{B} , this system is called the *Brunovsky canonical form* and it is always controllable from $\mathbf{u}(t)$.

Since (6.38) is said to be a *linearizing transformation* for the manipulator dynamic equation, one may invert this transformation to obtain the joint torque

$$\tau = H(q)T^+(q)(u - \ddot{r} - \dot{T}\dot{q}) + n(q, \dot{q}), \quad (6.41)$$

where T^+ denotes the *Moore–Penrose inverse* of the transformation matrix $T(q)$.

In the special case $\xi = e_q(t)$, and if we select $u(t)$ so that (6.39) is stable by the PD feedback $u = -K_P\xi - K_V\dot{\xi}$, then $T = -I_n$ and the control input torque $\tau(t)$ defined by (6.41) makes the manipulator move in such a way that $y(t)$ goes to zero. In this case, the feedback linearizing control and the inverse dynamics control become the same.

6.5.3 Passivity-Based Control

This method explicitly uses the passivity properties of the Lagrangian system [6.31,32]. In comparison with the inverse dynamics method, passivity-based controllers are expected to have better robust properties because they do not rely on the exact cancellation of the manipulator nonlinearities. The passivity-based control input is given by

$$\begin{aligned} \dot{q}_r &= \dot{q}_d + \alpha e_q, \quad \alpha > 0, \\ \tau &= H(q)\ddot{q}_r + C(q, \dot{q})\dot{q}_r + \tau_g(q) + K_V\dot{e}_q + K_P e_q. \end{aligned} \quad (6.42)$$

With (6.42), we obtain the following closed-loop system

$$H(q)\dot{s}_q + C(q, \dot{q})s_q + K_V\dot{e}_q + K_P e_q = 0, \quad (6.43)$$

where $s_q = \dot{e}_q + \alpha e_q$. Let us choose a Lyapunov function $V(y, t)$ as follows:

$$\begin{aligned} V &= \frac{1}{2}y^\top \begin{pmatrix} \alpha K_V + K_P + \alpha^2 H & \alpha H \\ \alpha H & H \end{pmatrix} y \\ &= \frac{1}{2}y^\top P y. \end{aligned} \quad (6.44)$$

Since the above equation is positive definite, it has a unique equilibrium at the origin, i.e., $y = (e_q^\top, \dot{e}_q^\top)^\top$

$= 0$. Moreover, V can be bounded by

$$\sigma_m \|y\|^2 \leq y^\top P y \leq \sigma_M \|y\|^2, \quad \sigma_M \geq \sigma_m > 0. \quad (6.45)$$

The time derivative of V gives

$$\dot{V} = -\dot{e}_q^\top K_V \dot{e}_q - \alpha e_q^\top K_P e_q = -y^\top Q y < 0, \quad (6.46)$$

where $Q = \text{diag}[\alpha K_P, K_V]$. Since Q is positive definite and quadratic in y , it can be also bounded by

$$\kappa_m \|y\|^2 \leq y^\top Q y \leq \kappa_M \|y\|^2, \quad \kappa_M \geq \kappa_m > 0. \quad (6.47)$$

Then, from the bound of the Lyapunov function V , we get

$$\dot{V} \leq -\kappa_m \|y\|^2 = -2\eta V, \quad \eta = \frac{\kappa_m}{\sigma_M} \quad (6.48)$$

which finally yields

$$V(t) \leq V(0)e^{-2\eta t}. \quad (6.49)$$

It has been shown that the value of α affects the tracking result dramatically [6.33]. The manipulator tends to vibrate for small values of α . Larger values of α allow better tracking performance and protect s_q from being spoiled by the velocity measurement noise when the position error is small. In [6.34], it was suggested that

$$K_P = \alpha K_V \quad (6.50)$$

be used for quadratic optimization.

6.5.4 Summary

In this section, we have reviewed some of the model-based motion control methods proposed to date. Under some control approaches, the closed-loop system theoretically, gives either asymptotic stability or globally exponential stability. However, such ideal performance cannot be obtained in practical implementation because factors such as sampling rate, measurement noise, disturbances, and unmodeled dynamics will limit the achievable gain and the performance of the control algorithms [6.33,35,36].

6.6 Computed-Torque Control

Through the years many kinds of robot control schemes have been proposed. Most of these can be considered as special cases of the class of computed-torque control (Fig. 6.5) which is the technique of applying feedback

linearization to nonlinear systems in general [6.37,38]. In the section, computed-torque control will be first introduced, and its variant, so-called computed-torque-like control, will be introduced later.

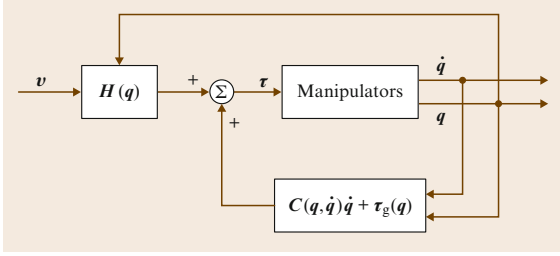


Fig. 6.5 Computed-torque control

6.6.1 Computed-Torque Control

Consider the control input (6.27)

$$\tau = H(q)v + C(q, \dot{q})\dot{q} + \tau_g(q),$$

which is also known as computed-torque control; it consists of an inner nonlinear compensation loop and an outer loop with an exogenous control signal v . Substituting this control law into the dynamical model of the robot manipulator, it follows that

$$\ddot{q} = v. \quad (6.51)$$

It is important to note that this control input converts a complicated nonlinear controller design problem into a simple design problem for a linear system consisting of n decoupled subsystems. One approach to the outer-loop control v is propositional-derivative (PD) feedback, as in (6.28):

$$v = \ddot{q}_d + K_V \dot{e}_q + K_P e_q,$$

in which case the overall control input becomes

$$\tau = H(q)(\ddot{q}_d + K_V \dot{e}_q + K_P e_q) + C(q, \dot{q})\dot{q} + \tau_g(q),$$

and the resulting linear error dynamics are

$$\ddot{e}_q + K_V \dot{e}_q + K_P e_q = 0. \quad (6.52)$$

According to linear system theory, convergence of the tracking error to zero is guaranteed [6.29, 39].

Remark.

- One usually lets K_V and K_P be $(n \times n)$ diagonal positive-definite gain matrices (i.e., $K_V = \text{diag}(k_{V,1}, \dots, k_{V,n}) > 0$, $K_P = \text{diag}(k_{P,1}, \dots, k_{P,n}) > 0$) to guarantee the stability of the error system. However, the format of the foregoing control never leads to *independent joint control* because the outer-loop multiplier $H(q)$ and the full nonlinear compensation term $C(q, \dot{q})\dot{q} + \tau_g(q)$ in the

inner loop scramble all joint signals among different control channels.

6.6.2 Computed-Torque-Like Control

It is worth noting that the implementation of computed-torque control requires that parameters of the dynamical model are accurately known and the control input is computed in real time. In order to avoid those problems, several variations of this control scheme have been proposed, for example, computed-torque-like control. An entire class of computed-torque-like controllers can be obtained by modifying the computed-torque control as

$$\tau = \hat{H}(q)v + \hat{C}(q, \dot{q})\dot{q} + \hat{\tau}_g(q), \quad (6.53)$$

where $\hat{\cdot}$ represents the computed or nominal value and indicates that the theoretically exact feedback linearization cannot be achieved in practice due to the uncertainty in the systems. The overall control scheme is depicted in Fig. 6.6.

Computed-Torque-Like Control with Variable-Structure Compensation

Since there is parametric uncertainty, compensation is required in the outer-loop design to achieve trajectory tracking. The following shows a computed-torque-like control with variable-structure compensation

$$v = \ddot{q}_d + K_V \dot{e}_q + K_P e_q + \Delta v, \quad (6.54)$$

where the variable-structure compensation Δv is devised as

$$\Delta v = \begin{cases} -\rho(x, t) \frac{B^T P x}{\|B^T P x\|} & \text{if } \|B^T P x\| \neq 0 \\ 0 & \text{if } \|B^T P x\| = 0, \end{cases} \quad (6.55)$$

where $x = (e_q^T, \dot{e}_q^T)^T$, $B = (0, I_n)^T$, P is a $(2n \times 2n)$ symmetric positive-definite matrix satisfying

$$PA + A^T P = -Q \quad (6.56)$$

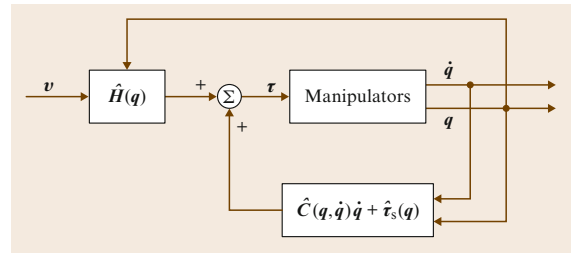


Fig. 6.6 Computed-torque-like control

with A being defined as

$$A = \begin{pmatrix} \mathbf{0} & \mathbf{I}_n \\ -\mathbf{K}_P & -\mathbf{K}_V \end{pmatrix}, \quad (6.57)$$

Q being any appropriate $(2n \times 2n)$ symmetric positive-definite matrix,

$$\rho(x, t) = \frac{1}{1-\alpha} [\alpha\beta + \|\mathbf{K}\| \|\mathbf{x}\| + \bar{\mathbf{H}}\phi(x, t)], \quad (6.58)$$

where α and β are positive constants such that $\|\mathbf{H}^{-1}(\mathbf{q})\hat{\mathbf{H}}(\mathbf{q}) - \mathbf{I}_n\| \leq \alpha < 1$ for all $\mathbf{q} \in R^n$ and $\sup_{t \in [0, \infty)} \|\dot{\mathbf{q}}_d(t)\| < \beta$, respectively, \mathbf{K} is the $(n \times 2n)$ matrix defined as $\mathbf{K} = [\mathbf{K}_P \ \mathbf{K}_V]$, $\bar{\lambda}_H$ being a positive constant such that $\|\mathbf{H}^{-1}(\mathbf{q})\| \leq \bar{\lambda}_H$ for all $\mathbf{q} \in R^n$, and the function ϕ being defined as

$$\begin{aligned} & \|[\hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})]\dot{\mathbf{q}} + [\hat{\boldsymbol{\tau}}_g(\mathbf{q}) - \boldsymbol{\tau}_g(\mathbf{q})]\| \\ & \leq \phi(x, t). \end{aligned} \quad (6.59)$$

Convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \mathbf{x}^\top \mathbf{P} \mathbf{x}, \quad (6.60)$$

following the stability analysis in [6.5, 40].

Remarks.

- By Property 6.1 in Sect. 6.1, there exist positive constants $\bar{\lambda}_H$ and $\bar{\lambda}_h$ such that $\bar{\lambda}_h \leq \|\mathbf{H}^{-1}(\mathbf{q})\| \leq \bar{\lambda}_H$ for all $\mathbf{q} \in R^n$. If we choose

$$\hat{\mathbf{H}} = \frac{1}{c} \mathbf{I}_n, \quad \text{where } c = \frac{\bar{\lambda}_H + \bar{\lambda}_h}{2}, \quad (6.61)$$

it can be shown that

$$\|\mathbf{H}^{-1}(\mathbf{q})\hat{\mathbf{H}}(\mathbf{q}) - \mathbf{I}_n\| \leq \frac{\bar{\lambda}_H - \bar{\lambda}_h}{\bar{\lambda}_H + \bar{\lambda}_h} \equiv \alpha < 1, \quad (6.62)$$

which indicates that there is always at least one choice of $\hat{\mathbf{H}}$ for some $\alpha < 1$.

- Due to the discontinuity in $\Delta \mathbf{v}$, chattering phenomenon may occur when the control scheme is applied. It is worth noting that chattering is often undesirable since the high-frequency component in the control can excite unmodeled dynamic effect (such as joint flexibility) [6.6, 29, 38]. In order to avoid chattering, the variable-structure compensation can be modified to become smooth, i. e.,

$$\Delta \mathbf{v} = \begin{cases} -\rho(x, t) \frac{\mathbf{B}^\top \mathbf{P} \mathbf{x}}{\|\mathbf{B}^\top \mathbf{P} \mathbf{x}\|} & \text{if } \|\mathbf{B}^\top \mathbf{P} \mathbf{x}\| > \varepsilon, \\ \frac{-\rho(x, t)}{\varepsilon} \mathbf{B}^\top \mathbf{P} \mathbf{x} & \text{if } \|\mathbf{B}^\top \mathbf{P} \mathbf{x}\| \leq \varepsilon, \end{cases} \quad (6.63)$$

where ε is a positive constant and is used as the boundary layer. Following this modification, convergence of tracking errors to a residual set can be ensured, and the size of this residual set can be made smaller by use of a smaller value of ε .

Computed-Torque-Like Control with Independent-Joint Compensation

Obviously, the previous compensation scheme is centralized, which implies that the online computation load is heavy and high-cost hardware is required for practical implementation. In order to solve this problem, a scheme with independent-joint compensation is introduced below. In this scheme, a computed-torque-like control is designed with estimates as

$$\hat{\mathbf{H}}(\mathbf{q}) = \mathbf{I}, \quad \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}, \quad \hat{\boldsymbol{\tau}}(\mathbf{q}) = \mathbf{0}. \quad (6.64)$$

Then, we use the outer-loop \mathbf{v} as

$$\mathbf{v} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \Delta \mathbf{v}, \quad (6.65)$$

where the positive constants \mathbf{K}_P and \mathbf{K}_V are selected to be sufficiently large, and the i -th component Δv_i of $\Delta \mathbf{v} = (\mathbf{v}_1, \dots, \mathbf{v}_n)^\top$ is

$$\Delta v_i = \begin{cases} -[\boldsymbol{\beta}^\top \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d)]^2 \frac{s_i}{\varepsilon_i} & \text{if } |s_i| \leq \frac{\varepsilon_i}{\boldsymbol{\beta}^\top \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d)}, \\ -\boldsymbol{\beta}^\top \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \frac{s_i}{|s_i|} & \text{if } |s_i| > \frac{\varepsilon_i}{\boldsymbol{\beta}^\top \mathbf{w}(\mathbf{q}_d, \dot{\mathbf{q}}_d)}. \end{cases} \quad (6.66)$$

In this compensation, $s_i = \dot{e}_{q,i} + \lambda_i e_{q,i}$, $i \in \{1, \dots, n\}$, and λ_i , $i \in \{1, \dots, n\}$ are positive constants. Furthermore, following the properties of robot manipulators, we have

$$\begin{aligned} & \|\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_d + \boldsymbol{\tau}_g(\mathbf{q})\| \\ & \leq \beta_1 + \beta_2 \|\mathbf{q}\| + \|\dot{\mathbf{q}}\| = \boldsymbol{\beta}^\top \mathbf{w}(\mathbf{q}, \dot{\mathbf{q}}) \end{aligned}$$

for some suitable positive constants β_1 , β_2 , and β_3 , where $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)^\top$ and

$$\mathbf{w}(\mathbf{q}, \dot{\mathbf{q}}) = [1, \|\mathbf{q}\|, \|\dot{\mathbf{q}}\|]^\top. \quad (6.67)$$

Finally, ε_i , $i \in \{1, \dots, n\}$, is the variable length of the boundary layer, satisfying

$$\dot{\varepsilon}_i = -g_i \varepsilon_i, \quad \varepsilon(0) > 0, \quad g_i > 0. \quad (6.68)$$

It is worth pointing out that the term \mathbf{w} in this control scheme is devised as the desired compensation rather than feedback. Furthermore, this scheme is in a form of independent-joint control and hence has the advantages introduced before. The convergence of the tracking error to zero can be shown using the Lyapunov function

$$V = \frac{1}{2} (\mathbf{e}_q^\top \dot{\mathbf{e}}_q^\top) \begin{pmatrix} \lambda \mathbf{K}_P & \mathbf{H} \\ \mathbf{H} & \lambda \mathbf{H} \end{pmatrix} \begin{pmatrix} \mathbf{e}_q \\ \dot{\mathbf{e}}_q \end{pmatrix} + \sum_{i=1}^n g_i^{-1} \varepsilon_i, \quad (6.69)$$

whose time derivative along the trajectory of the closed-loop systems follows

$$\dot{V} = -\alpha \left\| \begin{pmatrix} \mathbf{e}_q \\ \dot{\mathbf{e}}_q \end{pmatrix} \right\|^2 \quad (6.70)$$

with α being some positive constant, if sufficiently large \mathbf{K}_P and γ are used. The detailed analysis of stability, which requires Properties 6.3 and 6.4, can be found in [6.13].

Remark.

- Similar to computed-torque-like control with variable-structure compensation, we can consider the nonzero boundary layer as

$$\dot{\varepsilon}_i = -g_i \varepsilon_i, \quad \varepsilon(0) > 0, \quad g_i, \alpha_i > 0. \quad (6.71)$$

Following this modification, tracking errors converge to a residual set. The size of this residual set can be made smaller by the use of a smaller value of ε (i.e., smaller α_i).

For the task of point-to-point control, one PD controller with gravity compensation is designed with the estimates

$$\hat{\mathbf{H}}(\mathbf{q}) = \mathbf{I}, \quad \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}, \quad \hat{\boldsymbol{\tau}}_g(\mathbf{q}) = \boldsymbol{\tau}_g(\mathbf{q}) \quad (6.72)$$

with $\boldsymbol{\tau}_g(\mathbf{q})$ being the *gravity term* of the dynamical model of the robot manipulators. Then, we use the outer-loop \mathbf{v} as

$$\mathbf{v} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \quad (6.73)$$

such that the control input becomes

$$\boldsymbol{\tau} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q + \boldsymbol{\tau}_g(\mathbf{q}). \quad (6.74)$$

This scheme is much simpler to implement than the exact computed-torque controller. The convergence

of the tracking error to zero can be shown using the Lyapunov function

$$V = \frac{1}{2} \dot{\mathbf{e}}_q^\top \mathbf{H}(\mathbf{q}) \dot{\mathbf{e}}_q + \frac{1}{2} \mathbf{e}_q^\top \mathbf{K}_P \mathbf{e}_q, \quad (6.75)$$

whose time derivative along the solution trajectories of the closed-loop system is

$$\dot{V} = -\dot{\mathbf{e}}_q^\top \mathbf{K}_V \dot{\mathbf{e}}_q. \quad (6.76)$$

The detailed analysis of stability is given in [6.12]. It is necessary to note that this result is for the case of regulation, rather than for the case of tracking, since the former theoretical base, which relies on LaSalle's lemma, requires the system be autonomous (time invariant) [6.38, 41, 42].

Remark.

- If we neglect gravity in the dynamical model of the robot manipulators, then the gravity estimation can be omitted here, i.e., $\hat{\boldsymbol{\tau}}_g(\mathbf{q}) = \mathbf{0}$, so that the control law becomes

$$\boldsymbol{\tau} = \mathbf{v} = \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \quad (6.77)$$

which leads to pure PD control. The gain matrices, \mathbf{K}_P and \mathbf{K}_V , can be selected to be diagonal so that this PD control is in a form of independent-joint control developed based on the multijoint dynamic model.

6.6.3 Summary

In this section, we have presented two control schemes: the computed-torque and computed-torque-like control. The former transforms a multi-input multi-output (MIMO) nonlinear robotic system into a very simple decoupled linear closed-loop system whose control design is a well-established problem. Since the practical implementation of the former control requires preknowledge of all the manipulator parameters and its payload, which may not be realistic, the latter was introduced to relax the foregoing requirement and still to achieve the objective of tracking subject to system's uncertainty.

Further Reading

The PD control with different feedforward compensation for tracking control is investigated in [6.43]. An adaptive control scheme based on PD control is presented in [6.19].

6.7 Adaptive Control

An adaptive controller differs from an ordinary controller in that the controller parameters are time varying, and there is a mechanism for adjusting these parameters online based on some signals in the closed-loop systems. By the use of such control scheme, the control goal can be achieved even if there is parametric uncertainty in the plant. In this section, we will introduce several adaptive control schemes to deal with the case of imperfect knowledge of dynamical parameters of the robot manipulators. The control performance of those adaptive control schemes, including adaptive computed-torque control, adaptive inertia-related control, adaptive control based on passivity, and adaptive control with desired compensation, are basically derived from Property 6.5. Finally, the condition of persistent excitation, which is important in parameter convergence, will be addressed.

6.7.1 Adaptive Computed-Torque Control

The computed-torque control scheme is appealing, since it allows the designer to transform a MIMO highly coupled nonlinear system into a very simple decoupled linear system, whose control design is a well-established problem. However, this method of feedback linearization relies on perfect knowledge of the system parameters, and failure to have this will cause erroneous parametric estimates, resulting in a mismatch term in the closed-loop model of the error system. That term can be interpreted as a nonlinear perturbation acting at the input of the closed-loop system. In order to solve the problem due to parametric uncertainty, we instead consider the inverse dynamics method with parameter estimates of

$$\tau = \hat{H}(q)(\ddot{q}_d + K_V \dot{e}_q + K_P e_q) + \hat{C}(q, \dot{q})\dot{q} + \hat{\tau}_g(q), \quad (6.78)$$

where \hat{H} , \hat{C} , $\hat{\tau}_g$ have the same functional form as H , C , τ_g . From Property 6.5 of the dynamics model, we have

$$\hat{H}(q)\ddot{q} + \hat{C}(q, \dot{q})\dot{q} + \hat{\tau}_g(q) = Y(q, \dot{q}, \ddot{q})\hat{a}, \quad (6.79)$$

where $Y(q, \dot{q}, \ddot{q})$, called the regressor, is a known $(n \times r)$ function matrix and a is the $(r \times 1)$ vector that summarizes all the estimated parameters. Substituting this control input τ into the manipulator dynamics gives the following closed-loop error model

$$\hat{H}(q)(\ddot{e}_q + K_V \dot{e}_q + K_P e_q) = Y(q, \dot{q}, \ddot{q})\tilde{a}, \quad (6.80)$$

where $\tilde{a} = \hat{a} - a$. In order to acquire an appropriate adaptive law, we first assume that the acceleration term \ddot{q} is

measurable, and that the estimated inertia matrix $\hat{H}(q)$ is never singular. Now, for convenience, the error equation is rewritten as

$$\dot{x} = Ax + B\hat{H}^{-1}(q)Y(q, \dot{q}, \ddot{q})\tilde{a} \quad (6.81)$$

with $x = (e_q^\top, \dot{e}_q^\top)^\top$,

$$A = \begin{pmatrix} 0_n & I_n \\ -K_P & -K_V \end{pmatrix}, \quad B = \begin{pmatrix} 0_n \\ I_n \end{pmatrix}. \quad (6.82)$$

The adaptive law is considered as

$$\dot{\hat{a}} = -\Gamma^{-1}Y^\top(q, \dot{q}, \ddot{q})\hat{H}^{-1}(q)B^\top Px, \quad (6.83)$$

where Γ is an $(r \times r)$ positive-definite constant matrix, and P is a $(2n \times 2n)$ symmetric positive-definite constant matrix satisfying

$$PA + A^\top P = -Q, \quad (6.84)$$

with Q being a symmetric positive-definite constant matrix with coherent dimension. In this adaptive law, we made two assumptions:

- the joint acceleration \ddot{q} is measurable, and
- the bounded range of the unknown parameter is available.

The first assumption is to ensure that the regressor $Y(q, \dot{q}, \ddot{q})$ is known a priori, whereas the second assumption is to allow one to keep the estimate $\hat{H}(q)$ nonsingular by restricting the estimated parameter \hat{a} to lie within a range about the true parameter value.

Convergence of the tracking error and maintaining boundedness of all internal signals can actually be guaranteed by Lyapunov stability theory with the Lyapunov function

$$\dot{V} = -x^\top Qx. \quad (6.85)$$

Detailed stability analysis is given in [6.2].

Remark.

- For practical and theoretical reasons, the first assumption above is hardly acceptable. In most cases, it is not easy to obtain an accurate measure of acceleration; the robustness of the above adaptive control scheme with respect to such a disturbance has to be established. Moreover, from a pure theoretical viewpoint, measuring q, \dot{q}, \ddot{q} means that not only do we

need the whole system state vector, but we also need its derivative.

6.7.2 Adaptive Inertia-Related Control

Another adaptive control scheme is now introduced. This proposed scheme does not require the measurement of the manipulator's acceleration nor does it require inversion of the estimated inertia matrix. Hence, the drawbacks of the adaptive computed-torque control scheme are avoided. Let us consider the control input

$$\tau = \hat{H}(q)\dot{v} + \hat{C}(q, \dot{q})v + \hat{\tau}_g(q) + K_D s, \quad (6.86)$$

where the auxiliary signals v and s are defined as $v = \dot{q}_d + \Lambda e_q$ and $s = v - \dot{q} = \dot{e}_q + \Lambda e_q$, with Λ being an $(n \times n)$ positive-definite matrix. Following Property 6.5 of the dynamic model, we have

$$H(q)\dot{v} + C(q, \dot{q})v + \tau_g(q) = \bar{Y}(q, \dot{q}, v, \dot{v})a, \quad (6.87)$$

where $\bar{Y}(\cdot, \cdot, \cdot, \cdot)$ is an $(n \times r)$ matrix of known time functions. The formulation above is the same type of the parameter separation that was used in the formulation of the adaptive computed-torque control. Note that $\bar{Y}(q, \dot{q}, v, \dot{v})$ is independent of the joint acceleration. Similar to the formulation above, we also have

$$\hat{H}(q)\dot{v} + \hat{C}(q, \dot{q})v + \hat{\tau}_g(q) = \bar{Y}(q, \dot{q}, v, \dot{v})\hat{a}. \quad (6.88)$$

Substituting the control input into the equation of motion, it follows that

$$\begin{aligned} H(q)\ddot{q} + C(q, \dot{q})\dot{q} + \tau_g(q) \\ = \hat{H}(q)\dot{v} + \hat{C}(q, \dot{q})v + \hat{\tau}_g(q) + K_D s. \end{aligned}$$

Since $\ddot{q} = \dot{v} - \dot{s}$, $\dot{q} = v - s$, the previous result can be rewritten as

$$H(q)\dot{s} + C(q, \dot{q})s + K_D s = \bar{Y}(q, \dot{q}, v, \dot{v})\tilde{a}, \quad (6.89)$$

where $\tilde{a} = \hat{a} - a$. The adaptive law is considered as

$$\dot{\hat{a}} = -\Gamma \bar{Y}^\top(q, \dot{q}, v, \dot{v})s. \quad (6.90)$$

The convergence of the tracking error to zero with boundedness on all internal signals can be shown through Lyapunov stability theory using the following Lyapunov-like function

$$V = \frac{1}{2}s^\top H(q)s + \frac{1}{2}\tilde{a}^\top \Gamma^{-1}\tilde{a}, \quad (6.91)$$

whose time derivative along the trajectories of the closed-loop systems can be found to be

$$\dot{V} = -s^\top K_D s. \quad (6.92)$$

The detailed stability analysis is given in [6.32].

Remark.

- The restrictions for adaptive computed-torque control formerly seen have been removed here.
- The term $K_D s$ introduces a PD-type linear stabilizing control action to the error system model.
- The estimated parameters converge to the true parameters provided the reference trajectory satisfies the condition of persistency of excitation,

$$\begin{aligned} \alpha_1 I_r &\leq \int_{t_0}^{t_0+t} Y^\top(q_d, \dot{q}_d, v, \dot{v})Y(q_d, \dot{q}_d, v, \dot{v})dt \\ &\leq \alpha_2 I_r, \end{aligned}$$

for all t_0 , where α_1 , α_2 , and t are all positive constants.

6.7.3 Adaptive Control Based on Passivity

Taking a physics viewpoint of control, we see that the concept of passivity has become popular for the development of adaptive control schemes. Here, we illustrate how the concept of passivity can be used to design a class of adaptive control laws for robot manipulators. First, we define an auxiliary filtered tracking error signal r as

$$r = F^{-1}(s)e_q, \quad (6.93)$$

where

$$F^{-1}(s) = \left[sI_n + \frac{1}{s}K(s) \right] \quad (6.94)$$

and s is the Laplace transform variable. The $(n \times n)$ matrix $K(s)$ is chosen such that $F(s)$ is a strictly proper, stable transfer function matrix. As in the preceding schemes, the adaptive control strategies has close ties to the ability to separate the known functions from the unknown constant parameters. We use the expression given above to define

$$\begin{aligned} Z\phi &= H(q)[\ddot{q}_d + K(s)e_q] \\ &\quad + V(q, \dot{q}) \left[\dot{q}_n + \frac{1}{s}K(s)e_q \right] + \tau_g(q), \end{aligned}$$

where Z is a known $(n \times r)$ regression matrix and ϕ is a vector of unknown system parameters in the adaptive context. It is important to note that the above can be arranged such that Z and r do not depend on the measurement of the joint acceleration \ddot{q} . The adaptive control scheme given here is called the passivity approach because the mapping of $-r \rightarrow Z\phi$ is constructed to be

a passive mapping. That is, we develop an adaptive law such that

$$\int_0^t -\mathbf{r}^\top(\sigma)\mathbf{Z}(\sigma)\tilde{\boldsymbol{\phi}}(\sigma)d\sigma \geq -\beta \quad (6.95)$$

is satisfied for all time and for some positive scalar constant β . For this class of adaptive controllers, the control input is given by

$$\boldsymbol{\tau} = \mathbf{Z}\hat{\boldsymbol{\phi}} + \mathbf{K}_D \mathbf{r}, \quad (6.96)$$

Detailed analysis of stability is given in [6.44].

Remarks.

- If $\mathbf{K}(s)$ is selected such that $\mathbf{H}(s)$ has a relative degree of one, \mathbf{Z} and \mathbf{r} will not depend on $\ddot{\mathbf{q}}$.
- Many types of control schemes can be generated from the adaptive passive control approach by selected different transfer function matrices $\mathbf{K}(s)$ in the definition of \mathbf{r} .
- Note that, by defining $\mathbf{K}(s) = s\boldsymbol{\Lambda}$ such that $\mathbf{F}(s) = (s\mathbf{I}_n + \boldsymbol{\Lambda})^{-1}$, we have the control input

$$\boldsymbol{\tau} = \mathbf{Z}\hat{\boldsymbol{\phi}} - \mathbf{K}_D \mathbf{r},$$

with

$$\begin{aligned} \mathbf{Z}\hat{\boldsymbol{\phi}} &= \hat{\mathbf{H}}(\mathbf{q})(\ddot{\mathbf{q}}_d + \boldsymbol{\Lambda}\dot{\mathbf{e}}_q) + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})(\dot{\mathbf{q}}_d + \boldsymbol{\Lambda}\mathbf{e}_q) \\ &\quad + \hat{\boldsymbol{\tau}}_g(\mathbf{q}). \end{aligned}$$

The adaptive law may be chosen as

$$\dot{\hat{\boldsymbol{\phi}}} = \boldsymbol{\Gamma}\mathbf{Z}^\top(\dot{\mathbf{e}}_q + \boldsymbol{\Lambda}\mathbf{e}_q)$$

to satisfy the condition of passive mapping. This indicates that adaptive inertia-related control can be viewed as a special case of adaptive passive control.

6.7.4 Adaptive Control with Desired Compensation

In order to implement the adaptive control scheme, one needs to calculate the elements of $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})$ in real time. However, this procedure may be excessively time consuming since it involves computations with highly nonlinear functions of joint position and velocity. Consequently, the real-time implementation of such a scheme is rather difficult. To overcome this difficulty, the adaptive control with desired compensation was proposed and is discussed here. In other words, the variables \mathbf{q} , $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are replaced with the desired ones, namely, \mathbf{q}_d ,

$\dot{\mathbf{q}}_d$, and $\ddot{\mathbf{q}}_d$. Since the desired quantities are known in advance, all their corresponding calculations can be performed offline, which renders real-time implementation more plausible. Let us consider the control input

$$\boldsymbol{\tau} = \mathbf{Y}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)\hat{\mathbf{a}} + k_a \mathbf{s} + k_p \mathbf{e}_q + k_n \|\mathbf{e}_q\|^2 \mathbf{s}, \quad (6.97)$$

where the positive constants k_a , k_p , and k_n are sufficiently large, and the auxiliary signal \mathbf{s} is defined as $\mathbf{s} = \dot{\mathbf{e}}_q + \mathbf{e}_q$. The adaptive law is considered as

$$\dot{\hat{\mathbf{a}}} = -\boldsymbol{\Gamma}\mathbf{Y}^\top(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)\mathbf{s}. \quad (6.98)$$

It is worth noting that the desired compensation is adopted in both the control and adaptive laws such that the computational load can be drastically reduced. For the sake of this analysis, we note that

$$\begin{aligned} \|\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\mathbf{a} - \mathbf{Y}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)\hat{\mathbf{a}}\| \\ \leq \zeta_1 \|\mathbf{e}_q\| + \zeta_2 \|\mathbf{e}_q\|^2 + \zeta_3 \|\mathbf{s}\| + \zeta_4 \|\mathbf{s}\| \|\mathbf{e}_q\|, \end{aligned}$$

where ζ_1 , ζ_2 , ζ_3 , and ζ_4 are positive constants. In order to achieve trajectory tracking, it is required that

$$\begin{aligned} k_a &> \zeta_2 + \zeta_4, \\ k_p &> \frac{\zeta_1}{2} + \frac{\zeta_2}{4}, \\ k_v &> \frac{\zeta_1}{2} + \zeta_3 + \frac{\zeta_2}{4}, \end{aligned}$$

(i.e., the gains k_a , k_p , and k_v must be sufficiently large). The convergence of the tracking error to zero with boundedness on all internal signals can be proved through application of Lyapunov stability theory with the following Lyapunov-like function

$$V = \frac{1}{2} \mathbf{s}^\top \mathbf{H}(\mathbf{q}) \mathbf{s} + \frac{1}{2} k_p \mathbf{e}_q^\top \mathbf{e}_q + \frac{1}{2} \tilde{\mathbf{a}}^\top \boldsymbol{\Gamma}^{-1} \tilde{\mathbf{a}}, \quad (6.99)$$

whose time derivative along the trajectories of the closed-loop system can be derived as

$$\dot{V} \leq -\mathbf{x}^\top \mathbf{Q} \mathbf{x}, \quad (6.100)$$

where

$$\mathbf{x} = \begin{pmatrix} \|\mathbf{e}_q\| \\ \|\mathbf{s}\| \end{pmatrix}, \quad \mathbf{Q} = \begin{pmatrix} k_p - \zeta_2/4 & -\zeta_1/2 \\ -\zeta_1/2 & k_v - \zeta_3 - \zeta_4/4 \end{pmatrix}.$$

Detailed stability analysis can be found in [6.45].

6.7.5 Summary

Since the computed-torque control suffers from parametric uncertainty, a variety of adaptive control schemes

have been proposed. Firstly, we have presented an adaptive control scheme based on computed-torque control. Then, in order to overcome the mentioned drawbacks such as the measurability of the joint acceleration and the invertibility of the estimated inertia matrix, we presented an alternative adaptive control scheme that is free of these drawbacks. Recently, to incorporate a physics viewpoint into control, adaptive passivity-based control has become popular, and hence is introduced and discussed here. Finally, to reduce the computational load of the adaptive schemes, we presented an adaptive control with desired compensation.

Further Reading

A computationally very fast scheme dealing with adaptive control of rigid manipulators was presented in [6.46]. The stability analysis was completed by assuming that the joint dynamics are decoupled, i. e., that each joint is considered as an independent second-order linear system. Other pioneering works in the field can be found, for example, in [6.47, 48]; although none of the fundamental dynamic model properties are used, the complete dynamics are taken into account, but the control input is discontinuous and may lead to chattering. Positive definiteness of the inertia matrix is explicitly used in [6.49], although it was assumed that some time-varying quantities remain constant during the adap-

tation. It is interesting to note that all of these schemes were based on the concept of model reference adaptive control (MRAC) developed in [6.50] for linear systems. Therefore, they are conceptually very different from the truly nonlinear schemes presented in this section.

A passive-based modified version of the least-squares estimation scheme has been proposed in [6.51] and [6.52], which guaranteed closed-loop stability of the scheme. Other schemes can be found in [6.53], where no use is made of the skew-symmetry property, and in [6.54], where the recursive Newton–Euler formulations is used instead of the Lagrange formulation to derive the manipulator dynamics, and thus computation is simplified to facilitate practical implementation.

Even though adaptive control provides a solution to the problem of parametric uncertainty, the robustness of adaptive controllers remains a topic of great interest in the field. Indeed, measurement noise or unmodeled dynamics (e.g., flexibility) may result in unbounded closed-loop signals. In particular, the estimated parameters may diverge; this is a well-known phenomenon in adaptive control and is called parameter drift. Solutions inspired from the adaptive control of linear systems have been studied [6.55, 56], where a modified estimation ensures boundedness of the estimates. In [6.57], the controller in [6.32] is modified to enhance robustness.

6.8 Optimal and Robust Control

Given a nonlinear system, such as robotic manipulators, one can develop many stabilizing controls [6.29, 41]. In other words, the stability of the control system cannot determine a unique controller. It is natural that one seeks an optimal controller among the many stable ones. However, the design of an optimal controller is possible provided that a rather exact information on the target system is available, such as an exact system model [6.34, 58]. In the presence of discrepancy between the real system and its mathematical model, a designed optimal controller is no longer optimal, and may even end up being instable in the actual system. Generally speaking, the optimal control design framework is not the best one to deal with system uncertainty. To handle system uncertainty from the control design stage, a robust control design framework is necessary [6.59]. One of main objectives of robust control is to keep the controlled system stable even in presence of uncertainties in the mathematical model, unmodeled dynamics, and the like.

Let us consider an affine nonlinear system described by nonlinear time-varying differential equation in the state $\mathbf{x} = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{G}(\mathbf{x}, t)\mathbf{u} + \mathbf{P}(\mathbf{x}, t)\mathbf{w}, \quad (6.101)$$

where $\mathbf{u} \in \mathbb{R}^m$ is the control input, and $\mathbf{w} \in \mathbb{R}^w$ is the disturbance. Without disturbances or unmodeled dynamics, the system simplifies to

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, t) + \mathbf{G}(\mathbf{x}, t)\mathbf{u}. \quad (6.102)$$

Actually, there are many kinds of methods describing the nonlinear system according to the objective of control [6.1, 16, 21, 23, 34, 53].

6.8.1 Quadratic Optimal Control

Every optimal controller is based on its own cost function [6.60, 61]. One can define a cost function as [6.62, 63]

$$\mathbf{z} = \mathbf{H}(\mathbf{x}, t)\mathbf{x} + \mathbf{K}(\mathbf{x}, t)\mathbf{u}$$

such that $\mathbf{H}^\top(\mathbf{x}, t)\mathbf{K}(\mathbf{x}, t) = \mathbf{0}$, $\mathbf{K}^\top(\mathbf{x}, t)\mathbf{K}(\mathbf{x}, t) = \mathbf{R}(\mathbf{x}, t) > \mathbf{0}$, and $\mathbf{H}^\top(\mathbf{x}, t)\mathbf{H}(\mathbf{x}, t) = \mathbf{Q}(\mathbf{x}, t) > \mathbf{0}$. Then, we have

$$\frac{1}{2}\mathbf{z}^\top \mathbf{z} = \frac{1}{2}\mathbf{x}^\top \mathbf{Q}(\mathbf{x}, t)\mathbf{x} + \frac{1}{2}\mathbf{u}^\top \mathbf{R}(\mathbf{x}, t)\mathbf{u}.$$

The quadratic optimal control for the system (6.102) is found by solving, for a first order differentiable positive-definite function $V(\mathbf{x}, t)$, the Hamilton–Jacobi–Bellman (HJB) equation [6.34, 58]

$$\begin{aligned} 0 = \text{HJB}(\mathbf{x}, t; V) &= V_t(\mathbf{x}, t) + V_x(\mathbf{x}, t)\mathbf{f}(\mathbf{x}, t) \\ &\quad - \frac{1}{2}V_x(\mathbf{x}, t)\mathbf{G}(\mathbf{x}, t)\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^\top(\mathbf{x}, t)V_x^\top(\mathbf{x}, t) \\ &\quad + \frac{1}{2}\mathbf{Q}(\mathbf{x}, t), \end{aligned}$$

where $V_t = \frac{\partial V}{\partial t}$ and $V_x = \frac{\partial V}{\partial \mathbf{x}^\top}$. Then the quadratic optimal control is defined by

$$\mathbf{u} = -\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^\top(\mathbf{x}, t)V_x^\top(\mathbf{x}, t). \quad (6.103)$$

Note that the HJB equation is a nonlinear second-order partial differential equation in $V(\mathbf{x}, t)$.

Unlike the aforementioned optimal control problem, the so-called inverse quadratic optimal control problem is to find a set of $\mathbf{Q}(\mathbf{x}, t)$ and $\mathbf{R}(\mathbf{x}, t)$ for which the HJB equation has a solution $V(\mathbf{x}, t)$. Then the inverse quadratic optimal control is defined by (6.103).

6.8.2 Nonlinear \mathcal{H}_∞ Control

When disturbances are not negligible, one can deal with their effect such that

$$\int_0^t \mathbf{z}^\top(\mathbf{x}, \tau)\mathbf{z}(\mathbf{x}, \tau)d\tau \leq \gamma^2 \int_0^t \mathbf{w}^\top \mathbf{w} d\tau, \quad (6.104)$$

where $\gamma > 0$ specifies the L_2 -gain of the closed-loop system from the disturbance input \mathbf{w} to the cost variable \mathbf{z} . This is called the L_2 -gain attenuation requirement [6.62–64]. One systematic way to design an optimal and robust control is given by the nonlinear \mathcal{H}_∞ optimal control. Let $\gamma > 0$ be given, by solving the following equation:

$$\begin{aligned} \text{HJI}_\gamma(\mathbf{x}, t; V) &= V_t(\mathbf{x}, t) + V_x(\mathbf{x}, t)\mathbf{f}(\mathbf{x}, t) \\ &\quad - \frac{1}{2}V_x(\mathbf{x}, t)\{\mathbf{G}(\mathbf{x}, t)\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^\top(\mathbf{x}, t) \\ &\quad - \gamma^{-2}\mathbf{P}(\mathbf{x}, t)\mathbf{P}^\top(\mathbf{x}, t)\}V_x^\top(\mathbf{x}, t) \\ &\quad + \frac{1}{2}\mathbf{Q}(\mathbf{x}, t) \leq 0, \end{aligned} \quad (6.105)$$

then the control is defined by

$$\mathbf{u} = -\mathbf{R}^{-1}(\mathbf{x}, t)\mathbf{G}^\top(\mathbf{x}, t)V_x^\top(\mathbf{x}, t). \quad (6.106)$$

The partial differential inequality (6.105) is called the Hamilton–Jacobi–Isaac (HJI) inequality. Then one can define the inverse nonlinear \mathcal{H}_∞ optimal control problem that finds a set of $\mathbf{Q}(\mathbf{x}, t)$ and $\mathbf{R}(\mathbf{x}, t)$ such that the L_2 -gain requirement is achieved for a prescribed L_2 -gain γ [6.65].

Two things deserve further comment. The first is that the L_2 -gain requirement is only valid for disturbance signals \mathbf{w} whose L_2 -norm is bounded. The second is that \mathcal{H}_∞ optimal control is not uniquely defined. Hence, one can choose a quadratic optimal among many \mathcal{H}_∞ optimal controllers. Precisely speaking, the control (6.106) should be called \mathcal{H}_∞ suboptimal control, since the desired L_2 -gain is prescribed a priori. A true \mathcal{H}_∞ optimal control is to find a minimal value of γ , such that the L_2 -gain requirement is achieved.

6.8.3 Passivity-Based Design of Nonlinear \mathcal{H}_∞ Control

There are many methodologies for the design of optimal and/or robust controls. Among these, passivity-based controls can take full advantage of the properties described above [6.31]. They consist of two parts: one coming from the reference motion compensation while preserving the passivity of the system, and the other to achieve stability, robustness, and/or optimality [6.65, 66].

Let us suppose that the dynamic parameters are identified as $\hat{\mathbf{H}}(\mathbf{q})$, $\hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})$, and $\hat{\boldsymbol{\tau}}_g(\mathbf{q})$, whose counterparts are $\mathbf{H}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, and $\boldsymbol{\tau}_g(\mathbf{q})$, respectively. Then, passivity-based control generates the following tracking control laws

$$\boldsymbol{\tau} = \hat{\mathbf{H}}(\mathbf{q})\ddot{\mathbf{q}}_{\text{ref}} + \hat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_{\text{ref}} + \hat{\boldsymbol{\tau}}_g(\mathbf{q}) - \mathbf{u}, \quad (6.107)$$

where $\ddot{\mathbf{q}}_{\text{ref}}$ is the reference acceleration defined by

$$\ddot{\mathbf{q}}_{\text{ref}} = \ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \quad (6.108)$$

where $\mathbf{K}_V = \text{diag}\{k_{V,i}\} > 0$ and $\mathbf{K}_P = \text{diag}\{k_{P,i}\} > 0$. Two parameters are involved in generating the reference acceleration. Sometimes the following alternative method can be adopted

$$\ddot{\mathbf{q}}_{\text{ref}} = \ddot{\mathbf{q}}_d + \mathbf{K}_V \dot{\mathbf{e}}_q.$$

This reduces the order of the closed-loop system because the state $\mathbf{x} = (\mathbf{e}_q^\top, \dot{\mathbf{e}}_q^\top)^\top$ is sufficient for the system

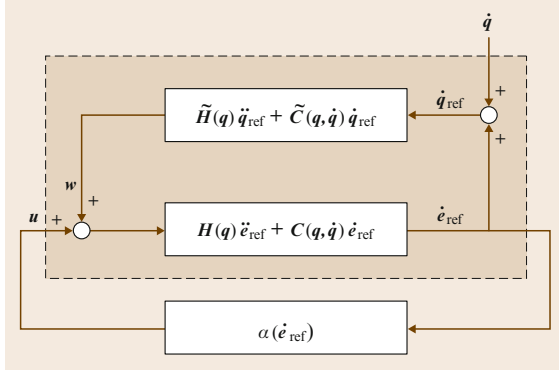


Fig. 6.7 The closed-loop system according to (6.109)

description, while the definition of (6.108) requires the state $\mathbf{x} = (\int \mathbf{e}_q^\top, \mathbf{e}_q^\top, \dot{\mathbf{e}}_q^\top)^\top$.

The closed-loop dynamics under the control is given by

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{e}}_{\text{ref}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{e}}_{\text{ref}} = \mathbf{u} + \mathbf{w}, \quad (6.109)$$

where

$$\begin{aligned} \ddot{\mathbf{e}}_{\text{ref}} &= \ddot{\mathbf{e}}_q + \mathbf{K}_V \dot{\mathbf{e}}_q + \mathbf{K}_P \mathbf{e}_q, \\ \dot{\mathbf{e}}_{\text{ref}} &= \dot{\mathbf{e}}_q + \mathbf{K}_V \mathbf{e}_q + \mathbf{K}_P \int \mathbf{e}_q. \end{aligned}$$

If $\mathbf{d}(t) = 0$ and $\hat{\mathbf{H}} = \mathbf{H}$, $\hat{\mathbf{C}} = \mathbf{C}$, $\hat{\boldsymbol{\tau}}_g = \boldsymbol{\tau}_g$, then $\mathbf{w} = 0$. Otherwise, the disturbance is defined as

$$\mathbf{w} = \tilde{\mathbf{H}}(\mathbf{q})\ddot{\mathbf{q}}_{\text{ref}} + \tilde{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}_{\text{ref}} + \tilde{\boldsymbol{\tau}}_g(\mathbf{q}) + \mathbf{d}(t), \quad (6.110)$$

where $\tilde{\mathbf{H}} = \mathbf{H} - \hat{\mathbf{H}}$, $\tilde{\mathbf{C}} = \mathbf{C} - \hat{\mathbf{C}}$, and $\tilde{\boldsymbol{\tau}}_g = \boldsymbol{\tau}_g - \hat{\boldsymbol{\tau}}_g$. It is of particular interest that the system (6.109) defines a passive mapping between $\mathbf{u} + \mathbf{w}$ and $\dot{\mathbf{e}}_{\text{ref}}$.

According to the manner in which the auxiliary control input \mathbf{u} is specified, passivity-based control can achieve stability, robustness, and/or optimality.

6.8.4 A Solution to Inverse Nonlinear \mathcal{H}_∞ Control

Let us define the auxiliary control input by the reference-error feedback

$$\mathbf{u} = -\alpha \mathbf{R}^{-1}(\mathbf{x}, t)\dot{\mathbf{e}}_{\text{ref}}, \quad (6.111)$$

where $\alpha > 1$ is arbitrary. Then, the control provides the inverse nonlinear \mathcal{H}_∞ optimality.

Theorem 6.1 Inverse Nonlinear \mathcal{H}_∞ Optimality [6.65]

Let the reference acceleration generation gain matrices \mathbf{K}_V and \mathbf{K}_P satisfy

$$\mathbf{K}_V^2 > 2\mathbf{K}_P. \quad (6.112)$$

Then for a given $\gamma > 0$, the reference error feedback

$$\mathbf{u} = -\mathbf{K}\dot{\mathbf{e}}_{\text{ref}} = -\mathbf{K}\left(\dot{\mathbf{e}}_q + \mathbf{K}_V \mathbf{e}_q + \mathbf{K}_P \int \mathbf{e}_q\right) \quad (6.113)$$

satisfies the L_2 -gain attenuation requirement for

$$\mathbf{Q} = \begin{pmatrix} \mathbf{K}_P^2 \mathbf{K}_\gamma & 0 & 0 \\ 0 & (\mathbf{K}_V^2 - 2\mathbf{K}_P)\mathbf{K}_\gamma & 0 \\ 0 & 0 & \mathbf{K}_\gamma \end{pmatrix}, \quad (6.114)$$

$$\mathbf{R} = \mathbf{K}^{-1}, \quad (6.115)$$

provided that

$$\mathbf{K}_\gamma = \mathbf{K} - \frac{1}{\gamma^2} \mathbf{I} > 0. \quad (6.116)$$

Given γ , one can set $\mathbf{K} = \alpha \frac{1}{\gamma^2} \mathbf{I}$ for $\alpha > 1$. This yields $\mathbf{K}_\gamma = (\alpha - 1) \frac{1}{\gamma^2} \mathbf{I}$.

When the inertia matrix is identified as a diagonal constant matrix such as $\hat{\mathbf{H}} = \text{diag}\{\hat{m}_i\}$, one should set $\hat{\mathbf{C}} = \mathbf{0}$. In addition, one can set $\hat{\boldsymbol{\tau}}_g = \mathbf{0}$. Then this results in a decoupled PID control of the form

$$\begin{aligned} \tau_i &= \hat{m}_i (\ddot{q}_{d,i} + k_{V,i} \dot{e}_{q,i} + k_{P,i} e_{q,i}) \\ &\quad + \alpha \frac{1}{\gamma^2} \left(\dot{e}_{q,i} + k_{V,i} e_{q,i} + k_{P,i} \int e_{q,i} \right) \end{aligned}$$

for $\alpha > 1$, which can be rewritten as

$$\begin{aligned} \tau_i &= \hat{m}_i \ddot{q}_{d,i} + \left(\hat{m}_i k_{V,i} + \alpha \frac{1}{\gamma^2} \right) \dot{e}_{q,i} \\ &\quad + \left(\hat{m}_i k_{P,i} + \alpha \frac{k_{V,i}}{\gamma^2} \right) e_{q,i} + \alpha \frac{k_{P,i}}{\gamma^2} \int e_{q,i}. \end{aligned} \quad (6.117)$$

This leads to a PID control with the desired acceleration feedforward [6.67] given by

$$\tau_i = \hat{m}_i \ddot{q}_{d,i} + k_{V,i}^* \dot{e}_{q,i} + k_{P,i}^* e_{q,i} + k_{I,i}^* \int e_{q,i}, \quad (6.118a)$$

where

$$k_{V,i}^* = \hat{m}_i k_{V,i} + \alpha \frac{1}{\gamma^2}, \quad (6.118b)$$

$$k_{P,i}^* = \hat{m}_i k_{P,i} + \alpha \frac{k_{V,i}}{\gamma^2}, \quad (6.118c)$$

$$k_{I,i}^* = \alpha \frac{k_{P,i}}{\gamma^2}. \quad (6.118d)$$

6.9 Digital Implementation

Most controllers introduced in the previous sections are digitally implemented on microprocessors. In this section basic but essential practical issues related to their computer implementation are discussed. When the controller is implemented on a computer control system, the analog inputs are read and the outputs are set with a certain sampling period. This is a drawback compared to analog implementations, since sampling introduces time delays into the control loop. Figure 6.8 shows the overall block diagram of control system with a boxed digital implementation part. When a digital computer is used to implement a control law, it is convenient to divide coding sequence in the interrupt routine into four process routines, as shown in Fig. 6.9. Reading the input signal from sensors and writing the control signal to digital-to-analog (D/A) converters synchronized at the correct frequency is very important. Therefore, these processes are located in the first routine. After saving counter values and extracting D/A values, which are already calculated one step before, the next routine produces reference values. Control routines with filters follow and produce scalar or vector control outputs. Finally, the user interface for checking parameter values is made and will be used for tuning and debugging.

6.9.1 Reference Trajectory Generation

Reference trajectory generation is a starting point and is used as a target value of control. It is a set of

values referred to the set point in every interrupt of the control processor and is therefore called the motion profile. Proper design of this profile is important because tracking and settling errors are very sensitive to it, since the target of the practical servo control is not regulating but tracking. A general trapezoidal shape profile is simple to calculate and constant current is required during acceleration and deceleration periods. However, large jerks occur at both ends of the motion, resulting in vibration. Large jerks (rapid change in acceleration, and therefore rapid changes in force) could cause substantial damage to the dynamic systems and induce unwanted vibrations, therefore a smoother profile is required, although this needs more calculation time. A simple motion profile using a third-order polynomial is as follows.

$$\begin{aligned} y(t) &= a + bt + ct^2 + dt^3, \\ \dot{y}(t) &= b + 2ct + 3dt^2, \\ \ddot{y}(t) &= 2c + 6dt. \end{aligned}$$

In practice, the motion profile equations are normalized and programmed as functions. With the constraints

$$\begin{aligned} y(0) &= 0, & y(1) &= 1, \\ \dot{y}(0) &= 0, & \dot{y}(1) &= 0, \end{aligned}$$

the program code takes the form shown in (6.119) when moving time and distance are normalized.

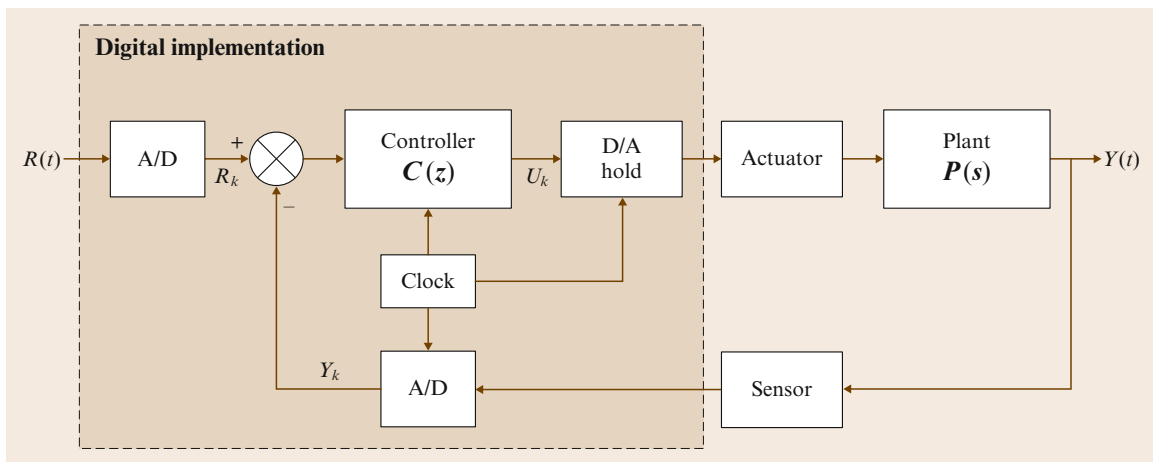


Fig. 6.8 Digital implementation of system control

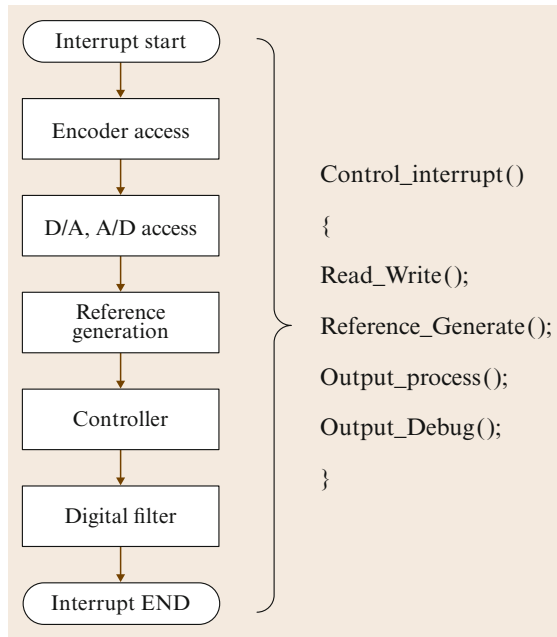


Fig. 6.9 The sequence in the interrupt routine for digital control

$$\begin{aligned}
 y\left(\frac{t}{T}\right) &= \left[3\left(\frac{t}{T}\right)^2 - 2\left(\frac{t}{T}\right)^3\right] S, \\
 \dot{y}\left(\frac{t}{T}\right) &= \left[\frac{6}{T}\left(\frac{t}{T}\right) - \frac{6}{T}\left(\frac{t}{T}\right)^2\right] S, \\
 \ddot{y}\left(\frac{t}{T}\right) &= \left[\frac{6}{T^2} - \frac{12}{T^2}\left(\frac{t}{T}\right)\right] S, \\
 \ddot{\ddot{y}}\left(\frac{t}{T}\right) &= -\frac{12}{T^3} S,
 \end{aligned} \tag{6.119}$$

where S is the distance and T is the time for motion.

For high-speed servo systems, the jerk profile is constant during runtime with large changes at start and stop. This means that high mechanical stiffness is required. Even though the calculation load increases, either fifth- or seventh-order polynomial profiles can be designed in order to satisfy acceleration or jerk constraints.

The motion profile is often saved in read-only memory (ROM) in the form of a table or located in the interrupt routine *Reference_Generate()* in the form of a function, as shown in Fig. 6.9. In most tracking servo control problem, a position profile is generated. However, velocity and acceleration are also calculated in the interrupt routine in real time to be used as inputs for the feedforward control. These are able to improve the

tracking performance, and the algorithm is as follows:

$$\begin{aligned}
 Pos_{\text{ref}} &= PosTable(pointer) \times Distance; \\
 Vel_{\text{ref}} &= Pos_{\text{ref}} - Pos_{\text{ref,OLD}} \\
 Acc_{\text{ref}} &= Vel_{\text{ref}} - Vel_{\text{ref,OLD}} \\
 &\vdots
 \end{aligned} \tag{6.120}$$

where Vel_{ref} and Acc_{ref} do not mean the real reference velocity and acceleration, but the reference velocity scaled by the interrupt time interval and the acceleration scaled by the square of the interrupt time interval, respectively, for practical use. Also, the limited size of the registers of a digital signal processor (DSP) or processor can cause errors due to the loss of significant digits when the velocity profile is calculated from the position profile using the simple difference formula in (6.120). This means that the velocity profile includes a periodic noise signal. This kind of numerical error creates real noise in the motion and reduces the performance of the control system when profiles of velocity, acceleration, and jerk are used as inputs for the feedforward control system for high-speed tracking.

Instead of the simple difference in (6.120), numerical differential equations can be used. Equation (6.121) shows the first-order backward-difference equations, which are the simplest form,

$$\begin{aligned}
 \dot{y}_k &= \frac{y_k - y_{k-1}}{\Delta t}, \\
 \ddot{y}_k &= \frac{y_k - 2y_{k-1} + y_{k-2}}{\Delta t^2}, \\
 \ddot{\ddot{y}}_k &= \frac{y_k - 3y_{k-1} + 3y_{k-2} - y_{k-3}}{\Delta t^3}.
 \end{aligned} \tag{6.121}$$

Saving each profiles in the form of a table can be another good method if the memory is large enough.

The important thing is that the maximum specification for the movement of controlled system, such as speed or acceleration or even jerk, should be considered in the profile design step. If the profile forces the system to breach the maximum specification, the possibility for the system to be driven recklessly is greatly increased due to missing of encoder pulses.

6.9.2 Z-Transform for Coding

Continuous-time systems are transformed into discrete-time systems by using the Z-transform. A discrete-time system is used to obtain a mathematical model that gives the behavior of a physical process at the sampling points, although the physical process is still a continuous-time

system. A Laplace transform is used for the analysis of control system in the s -domain. In most cases, the design of controllers and filters are done using tools in the s -domain. In order to realize those results in program code, understanding the Z -transform is essential. All controllers and filters designed in the s -domain can be easily translated to a program code through a Z -transform because it has the form of digitized difference sequences.

A coding **PID** controller is used as an example. In transfer-function form, this controller has the basic structure

$$\frac{Y(s)}{E(s)} = K_P + \frac{K_I}{s} + sK_V. \quad (6.122)$$

There are several methods for transformation from the frequency to the discrete domain. For stability conservation, backward Euler and Tustin algorithms are often used. Though the Tustin algorithm is known as the more exact one, the backward Euler algorithm is utilized in the following procedure.

After substituting the backward Euler equation into (6.122),

$$s \cong \frac{z-1}{zT}$$

the following discrete form is produced

$$\frac{Y(z)}{E(z)} = \frac{\alpha + \beta z^{-1} + \gamma z^{-2}}{T(1 - z^{-1})}, \quad (6.123)$$

where

$$\begin{aligned} \alpha &= K_I T^2 + K_P T + K_V, \\ \beta &= -K_P T - 2K_V, \\ \gamma &= K_V. \end{aligned}$$

Now, this is rearranged using the difference equation:

$$\begin{aligned} T(y_k - y_{k-1}) &= \alpha e_k + \beta e_{k-1} + \gamma e_{k-2}, \\ y_k &= y_{k-1} + \frac{1}{T}(\alpha e_k + \beta e_{k-1} + \gamma e_{k-2}), \\ y_k - y_{k-1} &= +\frac{\alpha}{T}e_k + \frac{\beta}{T}e_{k-1} + \frac{\gamma}{T}e_{k-2}. \end{aligned} \quad (6.124)$$

This can be directly coded in the program as follows:

$$\begin{aligned} e_k &= p_{k,\text{ref}} - p_k, \\ e_k^v &= v_{k,\text{ref}} - v_k \\ &= (p_{k,\text{ref}} - p_{k-1,\text{ref}}) - (p_k - p_{k-1}) \\ &= e_k - e_{k-1}, \\ \text{sum}_k &= \text{sum}_{k-1} + e_k = \sum_{j=0}^k e_k, \\ y_k &= K_{P,c}e_k + K_{V,c}(e_k - e_{k-1}) + K_{I,c}\text{sum}_k, \\ y_k - y_{k-1} &= (K_{P,c}e_k - K_{P,c}e_{k-1}) \\ &\quad + [K_{V,c}(e_k - e_{k-1})] - [K_{V,c}(e_{k-1} - e_{k-2})] \\ &\quad + (K_{I,c}\text{sum}_k - K_{I,c}\text{sum}_{k-1}) \\ &= (K_{P,c} + K_{V,c} + K_{I,c})e_k \\ &\quad - (K_{P,c} + 2K_{V,c})e_{k-1} + K_{V,c}e_{k-2}, \end{aligned} \quad (6.125)$$

where p_k is the present position, v_k is the present velocity, ref means reference, and c means code.

Comparing the parameters in (6.124) and (6.125), one obtains

$$\begin{aligned} \frac{\alpha}{T} &= K_P + \frac{K_V}{T} + K_I T = (K_{P,c} + K_{V,c} + K_{I,c}), \\ \frac{\beta}{T} &= -K_P - \frac{2K_V}{T} = -(K_{P,c} + 2K_{V,c}), \\ \frac{\gamma}{T} &= \frac{K_V}{T} = K_{V,c}, \end{aligned}$$

which shows that there is a relation between the designed and coded forms of the gains:

$$\begin{aligned} K_{P,c} &= K_P, \\ K_{V,c} &= \frac{K_V}{T}, \\ K_{I,c} &= K_I T. \end{aligned} \quad (6.126)$$

As the sampling frequency is increased in the same system, the coded K_V gain should be increased and the coded K_I gain should be decreased. Using this method, the designed controller can be coded in a DSP or micro-processor. However, sufficient analysis and simulation for control algorithms should be performed beforehand to obtain successful control system performance.

6.10 Learning Control

Since many robotic applications, such as pick-and-place operations, paintings, and circuit-board assembly, involve repetitive motions, it is natural to consider the use

of data gathered in previous cycles to try to improve the performance of the manipulator in subsequent cycles. This is the basic idea of repetitive control or learn-

ing control. Consider the robot model given in Sect. 6.1 and suppose that one is given a desired joint trajectory $\mathbf{q}_d(t)$ on a finite time interval $0 \leq t \leq T$. The reference trajectory \mathbf{q}_d is used in repeated trials of the manipulator, assuming either that the trajectory is periodic, $\mathbf{q}_d(T) = \mathbf{q}_d(0)$, (repetitive control) or that the robot is reinitialized to lie on the desired trajectory at the beginning of each trail (learning control). Hereafter, we use the term learning control to mean either repetitive or learning control.

6.10.1 Pure P-type Learning Control

Let $\boldsymbol{\tau}_k$ be the input torque during the k -th cycle, which produces an output $\mathbf{q}_k(t)$, $0 \leq t \leq T_{\text{bnd}}$. Now, let us consider the following set of assumptions:

- Assumption 1: Every trial ends at a fixed time of duration $T_{\text{bnd}} > 0$.
- Assumption 2: Repetition of the initial setting is satisfied.
- Assumption 3: Invariance of the system dynamics is ensured throughout these repeated trails.
- Assumption 4: Every output \mathbf{q}_k can be measured and thus the error signal $\Delta \mathbf{q}_k = \mathbf{q}_k - \mathbf{q}_d$ can be utilized in the construction of the next input $\boldsymbol{\tau}_{k+1}$.
- Assumption 5: The dynamics of the robot manipulators is invertible.

The learning control problem is to determine a recursive learning law \mathbf{L}

$$\boldsymbol{\tau}_{k+1} = \mathbf{L}[\boldsymbol{\tau}_k(t), \Delta \mathbf{q}_k(t)], \quad 0 \leq t \leq T_{\text{bnd}}, \quad (6.127)$$

where $\Delta \mathbf{q}_k(t) = \mathbf{q}_k(t) - \mathbf{q}_d(t)$, such that $\|\Delta \mathbf{q}_k\| \rightarrow 0$ as $k \rightarrow \infty$ in some suitably defined function norm, $\|\cdot\|$. The initial control input can be any control input that produces a stable output, such as PD control. Such learning control schemes are attractive because accurate models of the dynamics need not be known a priori.

Several approaches have been used to generate a suitable learning law \mathbf{L} and to prove convergence of the output error. A pure P-type learning law is one of the form

$$\boldsymbol{\tau}_{k+1}(t) = \boldsymbol{\tau}_k(t) - \Phi \Delta \mathbf{q}_k(t), \quad (6.128)$$

and is given this name because the correction term for the input torque at each iteration is proportional to the error $\Delta \mathbf{q}_k$. Now let $\boldsymbol{\tau}_d$ be defined by the computed-torque control, i. e.,

$$\begin{aligned} \boldsymbol{\tau}_d(t) = & \mathbf{H}[\mathbf{q}_d(t)]\ddot{\mathbf{q}}_d(t) + \mathbf{C}[\mathbf{q}_d(t), \dot{\mathbf{q}}_d(t)]\dot{\mathbf{q}}_d(t) \\ & + \boldsymbol{\tau}_g[\mathbf{q}_d(t)]. \end{aligned} \quad (6.129)$$

One should recall that the function $\boldsymbol{\tau}_k$ actually does not need to be computed; it is sufficient to know that it exists. Considering the P-type learning control law, we have

$$\Delta \boldsymbol{\tau}_{k+1}(t) = \Delta \boldsymbol{\tau}_k(t) - \Phi \Delta \mathbf{q}_k(t), \quad (6.130)$$

where $\Delta \boldsymbol{\tau}_k(t) = \boldsymbol{\tau}_k(t) - \boldsymbol{\tau}_d(t)$, so that

$$\|\Delta \boldsymbol{\tau}_{k+1}(t)\|^2 \leq \|\Delta \boldsymbol{\tau}_k(t)\|^2 - \beta \|\Phi \Delta \mathbf{q}_k(t)\|^2 \quad (6.131)$$

provided there exist positive constant λ and β such that

$$\int_0^{T_{\text{bnd}}} e^{-\lambda t} \Delta \mathbf{q}_k^\top \Delta \boldsymbol{\tau}_k(t) dt \geq \frac{1+\beta}{2} \|\Phi \Delta \mathbf{q}_k(t)\|^2 \quad (6.132)$$

for all k . It then follows from the inequality above that $\Delta \mathbf{q}_k \rightarrow 0$ in the norm sense as $k \rightarrow \infty$. Detailed stability analysis of this control scheme is given in [6.68, 69].

6.10.2 P-Type Learning Control with a Forgetting Factor

Although pure P-type learning control achieves the desired goal, several strict assumptions may be not valid in actual implementations, for example, there may be an initial setting error. Furthermore, there may be small but nonrepeatable fluctuations of dynamics. Finally, there may exist a (bounded) measurement noise $\boldsymbol{\xi}_k$ such that

$$\Delta \mathbf{q}_k(t) + \boldsymbol{\xi}_k(t) = [\mathbf{q}_k(t) + \boldsymbol{\xi}_k(t)] - \mathbf{q}_d(t). \quad (6.133)$$

Thus the learning control scheme may fail. In order to enhance the robustness of P-type learning control, a forgetting factor is introduced in the form

$$\begin{aligned} \boldsymbol{\tau}_{k+1}(t) = & (1 - \alpha)\boldsymbol{\tau}_k(t) + \alpha\boldsymbol{\tau}_0(t) \\ & - \Phi[\Delta \mathbf{q}_k(t) + \boldsymbol{\xi}_k(t)]. \end{aligned} \quad (6.134)$$

The original idea of using a forgetting factor in learning control originated with [6.70].

It has been rigorously proven that P-type learning control with a forgetting factor guarantees convergence to a neighborhood of the desired one of size $O(\alpha)$. Moreover, if the content of a long-term memory is refreshed after every k trials, where k is of $O(1/\alpha)$, then the trajectories converge to an ε -neighborhood of the desired control goal. The size of ε is dependent on the magnitude of the initial setting error, the nonrepeatable fluctuations of the dynamics, and the measurement noise. For a detailed stability investigation, please refer to [6.71, 72].

6.10.3 Summary

By applying learning control, the performance of repetitive tasks (such as painting or pick-and-place operation) is improved by utilizing data gathered in the previous cycles. In this section, two learning control schemes were introduced. First, pure P-type learning control and its ro-

business problem were described. Then P-type learning control with a forgetting factor was presented, enhancing the robustness of learning control.

Further Reading

Rigorous and refined exploration of learning control is first discussed independently in [6.2, 12].

References

- 6.1 C. Canudas de Wit, B. Siciliano, G. Bastin: *Theory of Robot Control* (Springer, London 1996)
- 6.2 J.J. Craig: *Adaptive Control of Mechanical Manipulators*. Ph.D. Thesis (UMI Dissertation Information Service, Ann Arbor 1986)
- 6.3 R.J. Schilling: *Fundamentals of Robotics: Analysis and Control* (Prentice-Hall, Upper Saddle River 1989)
- 6.4 L. Sciavicco, B. Siciliano: *Modeling and Control of Robot Manipulator* (McGraw-Hill, New York 1996)
- 6.5 M.W. Spong, M. Vidyasagar: *Robot Dynamics and Control* (Wiley, New York 1989)
- 6.6 M.W. Spong, F.L. Lewis, C.T. Abdallah (Eds.): *Robot Control* (IEEE, New York 1989)
- 6.7 C.H. An, C.G. Atkeson, J.M. Hollerbach: *Model-Based Control of a Robot Manipulator* (MIT Press, Cambridge, 1988)
- 6.8 R.M. Murray, Z. Xi, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC Press, Boca Raton 1994)
- 6.9 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)
- 6.10 O. Khatib: A unified approach for motion and force control of robot manipulators: The operational space formulation, *IEEE J. Robot. Autom.* **3**(1), 43–53 (1987)
- 6.11 J.Y.S. Luh, M.W. Walker, R.P.C. Paul: Resolved-acceleration control of mechanical manipulator, *IEEE Trans. Autom. Contr.* **25**(3), 468–474 (1980)
- 6.12 S. Arimoto, F. Miyazaki: Stability and robustness of PID feedback control for robot manipulators of sensory capability. In: *Robotics Research*, ed. by M. Brady, R. Paul (MIT Press, Cambridge 1984) pp.783–799
- 6.13 L.C. Fu: Robust adaptive decentralized control of robot manipulators, *IEEE Trans. Autom. Contr.* **37**(1), 106–110 (1992)
- 6.14 H. Seraji: Decentralized adaptive control of manipulators: Theory, simulation, and experimentation, *IEEE Trans. Robot. Autom.* **5**(2), 183–201 (1989)
- 6.15 J.G. Ziegler, N.B. Nichols: Optimum settings for automatic controllers, *ASME Trans.* **64**, 759–768 (1942)
- 6.16 Y. Choi, W.K. Chung: *PID Trajectory Tracking Control for Mechanical Systems*, Lecture Notes in Control and Information Sciences, Vol.289 (Springer, New York 2004)
- 6.17 R. Kelly: PD control with desired gravity compensation of robot manipulators: A review, *Int. J. Robot. Res.* **16**(5), 660–672 (1997)
- 6.18 M. Takegaki, S. Arimoto: A new feedback method for dynamic control of manipulators, *Trans. ASME J. Dyn. Syst. Meas. Contr.* **102**, 119–125 (1981)
- 6.19 P. Tomei: Adaptive PD controller for robot manipulators, *IEEE Trans. Robot. Autom.* **7**(4), 565–570 (1991)
- 6.20 R. Ortega, A. Loria, R. Kelly: A semi-globally stable output feedback PI^2D regulator for robot manipulators, *IEEE Trans. Autom. Contr.* **40**(8), 1432–1436 (1995)
- 6.21 D. Angeli: Input-to-State stability of PD-controlled robotic systems, *Automatica* **35**, 1285–1290 (1999)
- 6.22 J.A. Ramirez, I. Cervantes, R. Kelly: PID regulation of robot manipulators: stability and performance, *Sys. Contr. Lett.* **41**, 73–83 (2000)
- 6.23 R. Kelly: Global positioning of robot manipulators via PD control plus a class of nonlinear integral actions, *IEEE Trans. Autom. Contr.* **43**(7), 934–937 (1998)
- 6.24 Z. Qu, J. Dorsey: Robust tracking control of robots by a linear feedback law, *IEEE Trans. Autom. Contr.* **36**(9), 1081–1084 (1991)
- 6.25 H. Berghuis, H. Nijmeijer: Robust control of robots via linear estimated state feedback, *IEEE Trans. Autom. Contr.* **39**(10), 2159–2162 (1994)
- 6.26 Y. Choi, W.K. Chung, I.H. Suh: Performance and \mathcal{H}_∞ optimality of PID trajectory tracking controller for Lagrangian systems, *IEEE Trans. Robot. Autom.* **17**(6), 857–869 (2001)
- 6.27 K. Åström, T. Hägglund: *PID Controllers: Theory, Design, and Tuning* (Instrument Society of America, Research Triangle Park 1995)
- 6.28 C.C. Yu: *Autotuning of PID Controllers: Relay Feedback Approach* (Springer, London 1999)
- 6.29 F.L. Lewis, C.T. Abdallah, D.M. Dawson: *Control of Robot Manipulators* (Macmillan, New York 1993)
- 6.30 A. Isidori: *Nonlinear Control Systems: An Introduction*, Lecture Notes in Control and Information Sciences, Vol. 72 (Springer, New York 1985)
- 6.31 H. Berghuis, H. Nijmeijer: A passivity approach to controller-observer design for robots, *IEEE Trans. Robot. Autom.* **9**, 740–754 (1993)

- 6.32 J.J. Slotine, W. Li: On the adaptive control of robot manipulators, *Int. J. Robot. Res.* **6**(3), 49–59 (1987)
- 6.33 G. Liu, A.A. Goldenberg: Comparative study of robust saturation-based control of robot manipulators: analysis and experiments, *Int. J. Robot. Res.* **15**(5), 473–491 (1996)
- 6.34 D.M. Dawson, M. Grabbe, F.L. Lewis: Optimal control of a modified computed-torque controller for a robot manipulator, *Int. J. Robot. Autom.* **6**(3), 161–165 (1991)
- 6.35 D.M. Dawson, Z. Qu, J. Duffie: Robust tracking control for robot manipulators: theory, simulation and implementation, *Robotica* **11**, 201–208 (1993)
- 6.36 A. Jaritz, M.W. Spong: An experimental comparison of robust control algorithms on a direct drive manipulator, *IEEE Trans. Contr. Syst. Technol.* **4**(6), 627–640 (1996)
- 6.37 A. Isidori: *Nonlinear Control Systems*, 3rd edn. (Springer, New York 1995)
- 6.38 J.J. Slotine, W. Li: *Applied Nonlinear Control* (Prentice-Hall, Englewood Cliffs 1991)
- 6.39 W.J. Rugh: *Linear System Theory*, 2nd edn. (Prentice-Hall, Upper Saddle River 1996)
- 6.40 M.W. Spong, M. Vidyasagar: Robust microprocessor control of robot manipulators, *Automatica* **23**(3), 373–379 (1987)
- 6.41 H.K. Khalil: *Nonlinear Systems*, 3rd edn. (Prentice-Hall, Upper Saddle River 2002)
- 6.42 M. Vidyasagar: *Nonlinear Systems Analysis*, 2nd edn. (Prentice-Hall, Englewood Cliffs 1993)
- 6.43 J.T. Wen: A unified perspective on robot control: The energy Lyapunov function approach, *Int. J. Adapt. Contr. Signal Proc.* **4**, 487–500 (1990)
- 6.44 R. Ortega, M.W. Spong: Adaptive motion control of rigid robots: A tutorial, *Automatica* **25**(6), 877–888 (1989)
- 6.45 N. Sadeh, R. Horowitz: Stability and robustness analysis of a class of adaptive controllers for robotic manipulators, *Int. J. Robot. Res.* **9**(3), 74–92 (1990)
- 6.46 S. Dubowsky, D.T. DesForges: The application of model-reference adaptive control to robotic manipulators, *ASME J. Dyn. Syst. Meas. Control.* **37**(1), 106–110 (1992)
- 6.47 A. Balestrino, G. de Maria, L. Sciavicco: An adaptive model following control for robotic manipulators, *ASME J. Dyn. Syst. Meas. Control.* **105**, 143–151 (1983)
- 6.48 S. Nicosia, P. Tomei: Model reference adaptive control algorithms for industrial robots, *Automatica* **20**, 635–644 (1984)
- 6.49 R. Horowitz, M. Tomizuka: An adaptive control scheme for mechanical manipulators—Compensation of nonlinearity and decoupling control, *ASME J. Dyn. Syst. Meas. Contr.* **108**, 127–135 (1986)
- 6.50 I.D. Laudau: *Adaptive Control: The Model Reference Approach* (Dekker, New York 1979)
- 6.51 R. Lozano, C. Canudas de Wit: Passivity based adaptive control for mechanical manipulators using LS type estimation, *IEEE Trans. Autom. Contr.* **35**(12), 1363–1365 (1990)
- 6.52 B. Brogliato, I.D. Laudau, R. Lozano: Passive least squares type estimation algorithm for direct adaptive control, *Int. J. Adapt. Contr. Signal Process.* **6**, 35–44 (1992)
- 6.53 R. Johansson: Adaptive control of robot manipulator motion, *IEEE Trans. Robot. Autom.* **6**(4), 483–490 (1990)
- 6.54 M.W. Walker: Adaptive control of manipulators containing closed kinematic loops, *IEEE Trans. Robot. Autom.* **6**(1), 10–19 (1990)
- 6.55 J.S. Reed, P.A. Ioannou: Instability analysis and robust adaptive control of robotic manipulators, *IEEE Trans. Autom. Contr.* **5**(3), 74–92 (1989)
- 6.56 G. Tao: On robust adaptive control of robot manipulators, *Automatica* **28**(4), 803–807 (1992)
- 6.57 H. Berghuis, R. Ogata, H. Nijmeijer: A robust adaptive controller for robot manipulators, *Proc. IEEE Int. Conf. Robot. Autom.* (1992) pp.1876–1881
- 6.58 R. Johansson: Quadratic optimization of motion coordination and control, *IEEE Trans. Autom. Contr.* **35**(11), 1197–1208 (1990)
- 6.59 Z. Qu, D.M. Dawson: *Robust Tracking Control of Robot Manipulators* (IEEE, Piscataway 1996)
- 6.60 P. Dorato, C. Abdallah, V. Cerone: *Linear-Quadratic Control* (Prentice-Hall, Upper Saddle River 1995)
- 6.61 A. Locatelli: *Optimal Control: An Introduction* (Birkhäuser, Basel 2001)
- 6.62 A. Isidori: Feedback control of nonlinear systems, *Int. J. Robust Nonlin. Contr.* **2**, 291–311 (1992)
- 6.63 A.J. Van Der Schaft: Nonlinear state space \mathcal{H}_∞ control theory. In: *Essays on Control: Perspective in Theory and its Applications*, ed. by H.L. Trentelman, J.C. Willems (Birkhäuser, Basel 1993) pp.153–190
- 6.64 A.J. Van Der Schaft: L_2 -gain analysis of nonlinear systems and nonlinear state feedback \mathcal{H}_∞ control, *IEEE Trans. Autom. Contr.* **37**(6), 770–784 (1992)
- 6.65 J. Park, W.K. Chung, Y. Youm: Analytic nonlinear \mathcal{H}_∞ inverse-optimal control for Euler-Lagrange system, *IEEE Trans. Robot. Autom.* **16**(6), 847–854 (2000)
- 6.66 B.S. Chen, T.S. Lee, J.H. Feng: A nonlinear \mathcal{H}_∞ control design in robotics systems under parametric perturbation and external disturbance, *Int. J. Contr.* **59**(12), 439–461 (1994)
- 6.67 J. Park, W.K. Chung: Design of a robust \mathcal{H}_∞ PID control for industrial manipulators, *ASME J. Dyn. Syst. Meas. Contr.* **122**(4), 803–812 (2000)
- 6.68 S. Arimoto: Mathematical theory or learning with application to robot control. In: *Adaptive and Learning Control*, ed. by K.S. Narendra (Plenum, New York 1986) pp.379–388
- 6.69 S. Kawamura, F. Miyazaki, S. Arimoto: Realization of robot motion based on a learning method, *IEEE Trans. Syst. Man. Cybern.* **18**(1), 126–134 (1988)
- 6.70 G. Heinzinger, D. Frewick, B. Paden, F. Miyazaki: Robust learning control, *Proc. IEEE Int. Conf. Dec. Contr.* (1989)

- 6.71 S. Arimoto: Robustness of learning control for robot manipulators, Proc. IEEE Int. Conf. Dec. Contr. (Bellingham 1990) pp.1523–1528
- 6.72 S. Arimoto, T. Naiwa, H. Suzuki: Selective learning with a forgetting factor for robotic motion control, Proc. IEEE Int. Conf. Dec. Contr. (Bellingham 1991) pp.728–733

Force Control

7. Force Control

Luigi Villani, Joris De Schutter

A fundamental requirement for the success of a manipulation task is the capability to handle the physical contact between a robot and the environment. Pure motion control turns out to be inadequate because the unavoidable modeling errors and uncertainties may cause a rise of the contact force, ultimately leading to an unstable behavior during the interaction, especially in the presence of rigid environments. Force feedback and force control becomes mandatory to achieve a robust and versatile behavior of a robotic system in poorly structured environments as well as safe and dependable operation in the presence of humans. This chapter starts from the analysis of *indirect force control* strategies, conceived to keep the contact forces limited by ensuring a suitable compliant behavior to the end effector, without requiring an accurate model of the environment. Then the problem of *interaction tasks* modeling is analyzed, considering both the case of a rigid environment and the case of a compliant environment. For the specification of an interaction task, natural constraints set by the task geometry and artificial constraints set by the control strategy are established, with respect to suitable task frames. This formula-

7.1	Background	161
7.1.1	From Motion Control to Interaction Control	161
7.1.2	From Indirect Force Control to Hybrid Force/Motion Control	163
7.2	Indirect Force Control	164
7.2.1	Stiffness Control	164
7.2.2	Impedance Control	167
7.3	Interaction Tasks	171
7.3.1	Rigid Environment	171
7.3.2	Compliant Environment	173
7.3.3	Task Specification	174
7.3.4	Sensor-Based Contact Model Estimation	176
7.4	Hybrid Force/Motion Control	177
7.4.1	Acceleration-Resolved Approach	177
7.4.2	Passivity-Based Approach	179
7.4.3	Velocity-Resolved Approach	181
7.5	Conclusions and Further Reading	181
7.5.1	Indirect Force Control	182
7.5.2	Task Specification	182
7.5.3	Hybrid Force/Motion Control	182
	References	183

tion is the essential premise to the synthesis of *hybrid force/motion control* schemes.

7.1 Background

Research on robot force control has flourished in the past three decades. Such a wide interest is motivated by the general desire of providing robotic systems with enhanced sensory capabilities. Robots using force, touch, distance, and visual feedback are expected to autonomously operate in unstructured environments other than the typical industrial shop floor.

Since the early work on telemanipulation, the use of force feedback was conceived to assist the human op-

erator in the remote handling of objects with a slave manipulator. More recently, cooperative robot systems have been developed where two or more manipulators (viz. the fingers of a dexterous robot hand) are to be controlled so as to limit the exchanged forces and avoid squeezing of a commonly held object. Force control plays a fundamental role also in the achievement of robust and versatile behavior of robotic systems in open-ended environments, providing intelligent response in

unforeseen situations and enhancing human–robot interaction.

7.1.1 From Motion Control to Interaction Control

Control of the physical interaction between a robot manipulator and the environment is crucial for the successful execution of a number of practical tasks where the robot end-effector has to manipulate an object or perform some operation on a surface. Typical examples in industrial settings include polishing, deburring, machining or assembly. A complete classification of possible robot tasks, considering also nonindustrial applications, is practically infeasible in view of the large variety of cases that may occur, nor would such a classification be really useful to find a general strategy to control the interaction with the environment.

During contact, the environment may set constraints on the geometric paths that can be followed by the end-effector, denoted as *kinematic constraints*. This situation, corresponding to the contact with a stiff surface, is generally referred to as *constrained motion*. In other cases, the contact task is characterized by a dynamic interaction between the robot and the environment that can be inertial (as in pushing a block), dissipative (as in sliding on a surface with friction) or elastic (as in pushing against an elastically compliant wall). In all these cases, the use of a pure motion control strategy for controlling interaction is prone to failure, as explained below.

Successful execution of an interaction task with the environment by using motion control could be obtained only if the task were accurately planned. This would in turn require an accurate model of both the robot manipulator (kinematics and dynamics) and the environment (geometry and mechanical features). A manipulator model may be known with sufficient precision, but a detailed description of the environment is difficult to obtain.

To understand the importance of task planning accuracy, it is sufficient to observe that in order to perform a mechanical part mating with a positional approach the relative positioning of the parts should be guaranteed with an accuracy of an order of magnitude greater than part mechanical tolerance. Once the absolute position of one part is exactly known, the manipulator should guide the motion of the other with the same accuracy.

In practice, the planning errors may give rise to a contact force and moment, causing a deviation of the end-effector from the desired trajectory. On the other hand, the control system reacts to reduce such devia-

tions. This ultimately leads to a build-up of the contact force until saturation of the joint actuators is reached or breakage of the parts in contact occurs.

The higher the environment stiffness and position control accuracy are, the more easily a situation like the one just described can occur. This drawback can be overcome if a *compliant behavior* is ensured during the interaction. This compliant behavior can be achieved either in a passive or in an active fashion.

Passive Interaction Control

In passive interaction control the trajectory of the robot end-effector is modified by the interaction forces due to the inherent compliance of the robot. The compliance may be due to the structural compliance of the links, joints, and end-effector, or to the compliance of the position servo. Soft robot arms with elastic joints or links are purposely designed for intrinsically safe interaction with humans. In industrial applications, a mechanical device with passive compliance, known as the remote center of compliance (RCC) device [7.1], is widely adopted. An RCC is a compliant end-effector mounted on a rigid robot, designed and optimized for peg-into-hole assembly operations.

The passive approach to interaction control is very simple and cheap, because it does not require force/torque sensors; also, the preprogrammed trajectory of the end-effector must not be changed at execution time; moreover, the response of a passive compliance mechanism is much faster than active repositioning by a computer control algorithm. However, the use of passive compliance in industrial applications lacks flexibility, since for every robotic task a special-purpose compliant end-effector has to be designed and mounted. Also, it can only deal with small position and orientation deviations of the programmed trajectory. Finally, since no forces are measured, it can not guarantee that high contact forces will never occur.

Active Interaction Control

In active interaction control, the compliance of the robotic system is mainly ensured by a purposely designed control system. This approach usually requires the measurement of the contact force and moment, which are fed back to the controller and used to modify or even generate online the desired trajectory of the robot end-effector.

Active interaction control may overcome the aforementioned disadvantages of passive interaction control, but it is usually slower, more expensive, and more sophisticated. To obtain a reasonable task execution speed

and disturbance rejection capability, active interaction control has to be used in combination with some degree of passive compliance [7.2]: feedback, by definition, always comes after a motion and force error has occurred, hence some passive compliance is needed in order to keep the reaction forces below an acceptable threshold.

Force Measurements

For a general force-controlled task, six force components are required to provide complete contact force information: three translational force components and three torques. Often, a force/torque sensor is mounted at the robot wrist [7.3], but other possibilities exist, for example, force sensors can be placed on the fingertips of robotic hands [7.4]; also, external forces and moments can be estimated via shaft torque measurements of joint torque sensors [7.5, 6]. However, the majority of the applications of force control (including industrial applications) is concerned with wrist force/torque sensors. In this case, the weight and inertia of the tool mounted between the sensor and the environment (i.e., the robot end-effector) is assumed to be negligible or suitably compensated from the force/torque measurements. The force signals may be obtained using strain measurements, which results in a stiff sensor, or deformation measurements (e.g., optically), resulting in a compliant sensor. The latter approach has an advantage if additional passive compliance is desired.

7.1.2 From Indirect Force Control to Hybrid Force/Motion Control

Active interaction control strategies can be grouped into two categories: those performing indirect force control and those performing direct force control. The main difference between the two categories is that the former achieve force control via motion control, without explicit closure of a force feedback loop; the latter instead offer the possibility of controlling the contact force and moment to a desired value, thanks to the closure of a force feedback loop.

To the first category belongs *impedance control* (or *admittance control*) [7.7, 8], where the deviation of the end-effector motion from the desired motion due to the interaction with the environment is related to the contact force through a mechanical impedance/admittance with adjustable parameters. A robot manipulator under impedance (or admittance) control is described by an equivalent mass–spring–damper system with adjustable parameters. This relationship is an impedance if the robot control reacts to the motion deviation by gener-

ating forces, while it corresponds to an admittance if the robot control reacts to interaction forces by imposing a deviation from the desired motion. Special cases of impedance and admittance control are *stiffness control* and *compliance control* [7.9], respectively, where only the static relationship between the end-effector position and orientation deviation from the desired motion and the contact force and moment is considered. Notice that, in the robot control literature, the terms impedance control and admittance control are often used to refer to the same control scheme; the same happens for stiffness and compliance control. Moreover, if only the relationship between the contact force and moment and the end-effector linear and angular velocity is of interest, the corresponding control scheme is referred to as *damping control* [7.10].

Indirect force control schemes do not require, in principle, measurements of contact forces and moments; the resulting impedance or admittance is typically nonlinear and coupled. However, if a force/torque sensor is available, then force measurements can be used in the control scheme to achieve a linear and decoupled behavior.

Differently from indirect force control, direct force control requires an explicit model of the interaction task. In fact, the user has to specify the desired motion and the desired contact force and moment in a consistent way with respect to the constraints imposed by the environment. A widely adopted strategy belonging to this category is *hybrid force/motion control*, which aims at controlling the motion along the unconstrained task directions and force (and moment) along the constrained task directions. The starting point is the observation that, for many robotic tasks, it is possible to introduce an orthogonal reference frame, known as the compliance frame [7.11] (or task frame [7.12]) which allows one to specify the task in terms of natural and artificial constraints acting along and about the three orthogonal axes of this frame. Based on this decomposition, hybrid force/motion control allows simultaneous control of both the contact force and the end-effector motion in two mutually independent subspaces. Simple selection matrices acting on both the desired and feedback quantities serve this purpose for planar contact surfaces [7.13], whereas suitable projection matrices must be used for general contact tasks, which can also be derived from the explicit constraint equations [7.14–16]. Several implementations of hybrid motion control schemes are available, e.g., based on inverse dynamics control in the operational space [7.17], passivity-based control [7.18], or outer force control

loops closed around inner motion loops, typically available in industrial robots [7.2].

If an accurate model of the environment is not available, the force control action and the motion control action can be superimposed, resulting in a *parallel*

force/position control scheme. In this approach, the force controller is designed so as to dominate the motion controller; hence, a position error would be tolerated along the constrained task directions in order to ensure force regulation [7.19].

7.2 Indirect Force Control

To gain insight into the problems arising at the interaction between the end-effector of a robot manipulator and the environment, it is worth analyzing the effects of a motion control strategy in the presence of a contact force and moment. To this aim, assume that a reference frame Σ_e is attached to the end-effector, and let \mathbf{p}_e denote the position vector of the origin and \mathbf{R}_e the rotation matrix with respect to a fixed base frame. The end-effector velocity is denoted by the 6×1 twist vector $\mathbf{v}_e = (\dot{\mathbf{p}}_e^\top \ \boldsymbol{\omega}_e^\top)^\top$ where $\dot{\mathbf{p}}_e$ is the translational velocity and $\boldsymbol{\omega}_e$ the angular velocity, and can be computed from the $n \times 1$ joint velocity vector $\dot{\mathbf{q}}$ using the linear mapping

$$\mathbf{v}_e = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}}. \quad (7.1)$$

The matrix \mathbf{J} is the $6 \times n$ end-effector geometric Jacobian. For simplicity, the case of nonredundant nonsingular manipulators is considered; therefore, $n = 6$ and the Jacobian is a square nonsingular matrix. The force \mathbf{f}_e and moment \mathbf{m}_e applied by the end-effector to the environment are the components of the wrench $\mathbf{h}_e = (\mathbf{f}_e^\top \ \mathbf{m}_e^\top)^\top$.

It is useful to consider the operational space formulation of the dynamic model of a rigid robot manipulator in contact with the environment

$$\Lambda(\mathbf{q})\ddot{\mathbf{v}}_e + \Gamma(\mathbf{q}, \dot{\mathbf{q}})\mathbf{v}_e + \boldsymbol{\eta}(\mathbf{q}) = \mathbf{h}_c - \mathbf{h}_e, \quad (7.2)$$

where $\Lambda(\mathbf{q}) = (\mathbf{J}\mathbf{H}(\mathbf{q})^{-1}\mathbf{J}^\top)^{-1}$ is the 6×6 operational space inertia matrix, $\Gamma(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{J}^{-\top}\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{J}^{-1} - \Lambda(\mathbf{q})\dot{\mathbf{J}}\mathbf{J}^{-1}$ is the wrench including centrifugal and Coriolis effects, and $\boldsymbol{\eta}(\mathbf{q}) = \mathbf{J}^{-\top}\mathbf{g}(\mathbf{q})$ is the wrench of the gravitational effects; $\mathbf{H}(\mathbf{q})$, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ and $\mathbf{g}(\mathbf{q})$ are the corresponding quantities defined in the joint space. The vector $\mathbf{h}_c = \mathbf{J}^{-\top}\boldsymbol{\tau}$ is the equivalent end-effector wrench corresponding to the input joint torques $\boldsymbol{\tau}$.

7.2.1 Stiffness Control

In the classical operational space formulation, the end-effector position and orientation is described by a 6×1 vector $\mathbf{x}_e = (\mathbf{p}_e^\top \ \boldsymbol{\varphi}_e^\top)^\top$, where $\boldsymbol{\varphi}_e$ is a set of Euler angles

extracted from \mathbf{R}_e . Hence, a desired end-effector position and orientation can be assigned in terms of a vector \mathbf{x}_d , corresponding to the position of the origin \mathbf{p}_d and the rotation matrix \mathbf{R}_d of a desired frame Σ_d . The end-effector error can be denoted as $\Delta\mathbf{x}_{de} = \mathbf{x}_d - \mathbf{x}_e$, and the corresponding velocity error, assuming a constant \mathbf{x}_d , can be expressed as $\Delta\dot{\mathbf{x}}_{de} = -\dot{\mathbf{x}}_e = -\mathbf{A}^{-1}(\boldsymbol{\varphi}_e)\mathbf{v}_e$, with

$$\mathbf{A}(\boldsymbol{\varphi}_e) = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}(\boldsymbol{\varphi}_e) \end{pmatrix},$$

where \mathbf{I} is the 3×3 identity matrix, $\mathbf{0}$ is a 3×3 null matrix, and \mathbf{T} is the 3×3 matrix of the mapping $\boldsymbol{\omega}_e = \mathbf{T}(\boldsymbol{\varphi}_e)\dot{\boldsymbol{\varphi}}_e$, depending on the particular choice of the Euler angles.

Consider the motion control law

$$\mathbf{h}_c = \mathbf{A}^{-\top}(\boldsymbol{\varphi}_e)\mathbf{K}_P\Delta\mathbf{x}_{de} - \mathbf{K}_D\mathbf{v}_e + \boldsymbol{\eta}(\mathbf{q}), \quad (7.3)$$

corresponding to a simple PD + gravity compensation control in the operational space, where \mathbf{K}_P and \mathbf{K}_D are symmetric and positive-definite 6×6 matrices.

In the absence of interaction with the environment (i.e., when $\mathbf{h}_e = \mathbf{0}$), the equilibrium $\mathbf{v}_e = \mathbf{0}$, $\Delta\mathbf{x}_{de} = \mathbf{0}$ for the closed-loop system, corresponding to the desired position and orientation for the end-effector, is asymptotically stable. The stability proof is based on the positive-definite Lyapunov function

$$V = \frac{1}{2}\mathbf{v}_e^\top \Lambda(\mathbf{q})\mathbf{v}_e + \frac{1}{2}\Delta\mathbf{x}_{de}^\top \mathbf{K}_P\Delta\mathbf{x}_{de},$$

whose time derivative along the trajectories of the closed-loop system is the negative semidefinite function

$$\dot{V} = -\mathbf{v}_e^\top \mathbf{K}_D\mathbf{v}_e. \quad (7.4)$$

In the presence of a constant wrench \mathbf{h}_e , using a similar Lyapunov argument, a different asymptotically stable equilibrium can be found, with a nonnull $\Delta\mathbf{x}_{de}$. The new equilibrium is the solution of the equation

$$\mathbf{A}^{-\top}(\boldsymbol{\varphi}_e)\mathbf{K}_P\Delta\mathbf{x}_{de} - \mathbf{h}_e = \mathbf{0},$$

which can be written in the form

$$\Delta\mathbf{x}_{de} = \mathbf{K}_P^{-1}\mathbf{A}^\top(\boldsymbol{\varphi}_e)\mathbf{h}_e, \quad (7.5)$$

or, equivalently, as

$$\mathbf{h}_e = \mathbf{A}^{-\top}(\boldsymbol{\varphi}_e) \mathbf{K}_P \Delta \mathbf{x}_{de}. \quad (7.6)$$

Equation (7.6) shows that in the steady state the end-effector, under a proportional control action on the position and orientation error, behaves as a six-degree-of-freedom (DOF) spring in respect of the external force and moment \mathbf{h}_e . Thus, the matrix \mathbf{K}_P plays the role of an *active stiffness*, meaning that it is possible to act on the elements of \mathbf{K}_P so as to ensure a suitable elastic behavior of the end-effector during the interaction. Analogously, (7.5) represents a compliance relationship, where the matrix \mathbf{K}_P^{-1} plays the role of an *active compliance*. This approach, consisting of assigning a desired position and orientation and a suitable static relationship between the deviation of the end-effector position and orientation from the desired motion and the force exerted on the environment, is known as *stiffness control*.

The selection of the stiffness/compliance parameters is not easy, and strongly depends on the task to be executed. A higher value of the active stiffness means a higher accuracy of the position control at the expense of higher interaction forces. Hence, if it is expected to meet some physical constraint in a particular direction, the end-effector stiffness in that direction should be made low to ensure low interaction forces. Conversely, along the directions where physical constraints are not expected, the end-effector stiffness should be made high so as to follow closely the desired position. This allows discrepancies between the desired and achievable positions due to the constraints imposed by the environment to be resolved without excessive contact forces and moments.

It must be pointed out, however, that a selective stiffness behavior along different directions cannot be effectively assigned in practice on the basis of (7.6). This can easily be understood by using the classical definition of a mechanical stiffness for two bodies connected by a 6-DOF spring, in terms of the linear mapping between the infinitesimal twist displacement of the two bodies at an unloaded equilibrium and the elastic wrench.

In the case of the active stiffness, the two bodies are, respectively, the end-effector, with the attached frame Σ_e , and a virtual body, attached to the desired frame Σ_d . Hence, from (7.6), the following mapping can be derived

$$\mathbf{h}_e = \mathbf{A}^{-\top}(\boldsymbol{\varphi}_e) \mathbf{K}_P \mathbf{A}^{-1}(\boldsymbol{\varphi}_e) \delta \mathbf{x}_{de}, \quad (7.7)$$

in the case of an infinitesimal twist displacement $\delta \mathbf{x}_{de}$ defined as

$$\delta \mathbf{x}_{de} = \begin{pmatrix} \delta \mathbf{p}_{de} \\ \delta \boldsymbol{\theta}_{de} \end{pmatrix} = \begin{pmatrix} \Delta \dot{\mathbf{p}}_{de} \\ \Delta \dot{\boldsymbol{\omega}}_{de} \end{pmatrix} dt = - \begin{pmatrix} \dot{\mathbf{p}}_e \\ \dot{\boldsymbol{\omega}}_e \end{pmatrix} dt,$$

where $\Delta \dot{\mathbf{p}}_{de} = \dot{\mathbf{p}}_d - \dot{\mathbf{p}}_e$ is the time derivative of the position error $\Delta \mathbf{p}_{de} = \mathbf{p}_d - \mathbf{p}_e$ and $\Delta \dot{\boldsymbol{\omega}}_{de} = \dot{\boldsymbol{\omega}}_d - \dot{\boldsymbol{\omega}}_e$ is the angular velocity error. Equation (7.7) shows that the actual stiffness matrix is $\mathbf{A}^{-\top}(\boldsymbol{\varphi}_e) \mathbf{K}_P \mathbf{A}^{-1}(\boldsymbol{\varphi}_e)$, which depends on the end-effector orientation through the vector $\boldsymbol{\varphi}_e$, so that, in practice, the selection of the stiffness parameters is quite difficult.

This problem can be overcome by defining a *geometrically consistent active stiffness*, with the same structure and properties as ideal mechanical springs.

Mechanical Springs

Consider two elastically coupled rigid bodies A and B and two reference frames Σ_a and Σ_b , attached to A and B , respectively. Assuming that at equilibrium frames Σ_a and Σ_b coincide, the compliant behavior near the equilibrium can be described by the linear mapping

$$\mathbf{h}_b^b = \mathbf{K} \delta \mathbf{x}_{ab}^b = \begin{pmatrix} \mathbf{K}_t & \mathbf{K}_c \\ \mathbf{K}_c^\top & \mathbf{K}_o \end{pmatrix} \delta \mathbf{x}_{ab}^b, \quad (7.8)$$

where \mathbf{h}_b^b is the elastic wrench applied to body B , expressed in frame B , in the presence of an infinitesimal twist displacement $\delta \mathbf{x}_{ab}^b$ of frame Σ_a with respect to frame Σ_b , expressed in frame B . The elastic wrench and the infinitesimal twist displacement in (7.8) can also be expressed equivalently in frame Σ_a , since Σ_a and Σ_b coincide at equilibrium. Therefore, $\mathbf{h}_b^b = \mathbf{h}_b^a$ and $\delta \mathbf{x}_{ab}^b = \delta \mathbf{x}_{ab}^a$; moreover, for the elastic wrench applied to body A , $\mathbf{h}_a^a = \mathbf{K}_t \delta \mathbf{x}_{ba}^a = -\mathbf{h}_b^b$ being $\delta \mathbf{x}_{ba}^a = -\delta \mathbf{x}_{ab}^b$. This property of the mapping (7.8) is known as *port symmetry*.

In (7.8), \mathbf{K} is the 6×6 symmetric positive-semidefinite stiffness matrix. The 3×3 matrices \mathbf{K}_t and \mathbf{K}_o , called respectively the *translational stiffness* and *rotational stiffness*, are also symmetric. It can be shown that, if the 3×3 matrix \mathbf{K}_c , called the *coupling stiffness* is symmetric, there is maximum decoupling between rotation and translation. In this case, the point corresponding to the coinciding origins of the frames Σ_a and Σ_b is called the *center of stiffness*. Similar definitions and results can be formulated for the case of the compliance matrix $\mathbf{C} = \mathbf{K}^{-1}$. In particular, it is possible to define a *center of compliance* in the case that the off-diagonal blocks of the compliance matrix are symmetric. The center of stiffness and compliance do not necessarily coincide.

There are special cases in which no coupling exists between translation and rotation, i.e., a relative transla-

tion of the bodies results in a wrench corresponding to a pure force along an axis through the center of stiffness; also, a relative rotation of the bodies results in a wrench that is equivalent to a pure torque about an axis through the centers of stiffness. In these cases, the center of stiffness and compliance coincide. Mechanical systems with completely decoupled behavior are, e.g., the remote center of compliance (RCC) devices.

Since \mathbf{K}_t is symmetric, there exists a rotation matrix \mathbf{R}_t with respect to the frame $\Sigma_a = \Sigma_b$ at equilibrium, such that $\mathbf{K}_t = \mathbf{R}_t \mathbf{\Gamma}_t \mathbf{R}_t^\top$, and $\mathbf{\Gamma}_t$ is a diagonal matrix whose diagonal elements are the *principal translational stiffnesses* in the directions corresponding to the columns of the rotation matrix \mathbf{R}_t , known as the *principal axes of translational stiffness*. Analogously, \mathbf{K}_o can be expressed as $\mathbf{K}_o = \mathbf{R}_o \mathbf{\Gamma}_o \mathbf{R}_o^\top$, where the diagonal elements of $\mathbf{\Gamma}_o$ are the *principal rotational stiffnesses* about the axes corresponding to the columns of rotation matrix \mathbf{R}_o , known as the *principal axes of rotational stiffness*. Moreover, assuming that the origins of Σ_a and Σ_b at equilibrium coincide with the center of stiffness, the expression $\mathbf{K}_c = \mathbf{R}_c \mathbf{\Gamma}_c \mathbf{R}_c^\top$ can be found, where the diagonal elements of $\mathbf{\Gamma}_c$ are the *principal coupling stiffnesses* along the directions corresponding to the columns of the rotation matrix \mathbf{R}_c , known as the *principal axes of coupling stiffness*. In sum, a 6×6 stiffness matrix can be specified, with respect to a frame with origin in the center of stiffness, in terms of the principal stiffness parameters and principal axes.

Notice that the mechanical stiffness defined by (7.8) describes the behavior of an ideal 6-DOF spring which stores potential energy. The potential energy function of an ideal stiffness depends only on the relative position and orientation of the two attached bodies and is port symmetric. A physical 6-DOF spring has a predominant behavior similar to the ideal one, but nevertheless it always has parasitic effects causing energy dissipation.

Geometrically Consistent Active Stiffness

To achieve a geometrically consistent 6-DOF active stiffness, a suitable definition of the proportional control action in control law (7.3) is required. This control action can be interpreted as the elastic wrench applied to the end-effector, in the presence of a *finite displacement* of the desired frame Σ_d with respect to the end-effector frame Σ_e . Hence, the properties of the ideal mechanical stiffness for small displacements should be extended to the case of finite displacements. Moreover, to guarantee asymptotic stability in the sense of Lyapunov, a suitable potential elastic energy function must be defined.

For simplicity, it is assumed that the coupling stiffness matrix is zero. Hence, the potential elastic energy can be computed as the sum of a translational potential energy and a rotational potential energy.

The translational potential energy can be defined as

$$V_t = \frac{1}{2} \Delta \mathbf{p}_{de}^\top \mathbf{K}'_{Pt} \Delta \mathbf{p}_{de} \quad (7.9)$$

with

$$\mathbf{K}'_{Pt} = \frac{1}{2} \mathbf{R}_d \mathbf{K}_{Pt} \mathbf{R}_d^\top + \frac{1}{2} \mathbf{R}_e \mathbf{K}_{Pt} \mathbf{R}_e^\top,$$

where \mathbf{K}_{Pt} is a 3×3 symmetric positive-definite matrix. The use of \mathbf{K}'_{Pt} in lieu of \mathbf{K}_{Pt} in (7.9) guarantees that the potential energy is port symmetric also in the case of finite displacements. Matrices \mathbf{K}'_{Pt} and \mathbf{K}_{Pt} coincide at equilibrium (i.e., when $\mathbf{R}_d = \mathbf{R}_e$) and in the case of isotropic translational stiffness (i.e., when $\mathbf{K}_{Pt} = k_{Pt} \mathbf{I}$).

The computation of the power \dot{V}_t yields

$$\dot{V}_t = \Delta \dot{\mathbf{p}}_{de}^\top \mathbf{f}_{\Delta t}^e + \Delta \omega_{de}^\top \mathbf{m}_{\Delta t}^e,$$

where $\Delta \dot{\mathbf{p}}_{de}^e$ is the time derivative of the position displacement $\Delta \mathbf{p}_{de}^e = \mathbf{R}_e^\top (\mathbf{p}_d - \mathbf{p}_e)$, while $\Delta \omega_{de}^e = \mathbf{R}_e^\top (\boldsymbol{\omega}_d - \boldsymbol{\omega}_e)$. The vectors $\mathbf{f}_{\Delta t}^e$ and $\mathbf{m}_{\Delta t}^e$ are, respectively, the elastic force and moment applied to the end-effector in the presence of the finite position displacement $\Delta \mathbf{p}_{de}^e$. These vectors have the following expressions when computed in the base frame

$$\mathbf{f}_{\Delta t} = \mathbf{K}'_{Pt} \Delta \mathbf{p}_{de} \quad \mathbf{m}_{\Delta t} = \mathbf{K}''_{Pt} \Delta \mathbf{p}_{de} \quad (7.10)$$

with

$$\mathbf{K}''_{Pt} = \frac{1}{2} S(\Delta \mathbf{p}_{de}) \mathbf{R}_d \mathbf{K}_{Pt} \mathbf{R}_d^\top,$$

where $S(\cdot)$ is the skew-symmetric operator performing the vector product. The vector $\mathbf{h}_{\Delta t} = (\mathbf{f}_{\Delta t}^\top \mathbf{m}_{\Delta t}^\top)^\top$ is the elastic wrench applied to the end-effector in the presence of a finite position displacement $\Delta \mathbf{p}_{de}$ and a null orientation displacement. The moment $\mathbf{m}_{\Delta t}$ is null in the case of isotropic translational stiffness.

To define the rotational potential energy, a suitable definition of the orientation displacement between the frames Σ_d and Σ_e has to be adopted. A possible choice is the vector part of the unit quaternion $\{\eta_{de}, \epsilon_{de}^e\}$ that can be extracted from matrix $\mathbf{R}_d^e = \mathbf{R}_e^\top \mathbf{R}_d$. Hence, the orientation potential energy has the form

$$V_o = 2\epsilon_{de}^\top \mathbf{K}_{Po} \epsilon_{de}^e, \quad (7.11)$$

where \mathbf{K}_{Po} is a 3×3 symmetric positive-definite matrix. The function V_o is port symmetric because $\epsilon_{de}^e = -\epsilon_{ed}^d$.

The computation of the power \dot{V}_0 yields

$$\dot{V}_0 = \Delta \omega_{de}^{\top} \mathbf{m}_{\Delta o}^e,$$

where

$$\mathbf{m}_{\Delta o} = \mathbf{K}'_{Po} \epsilon_{de}, \quad (7.12)$$

with

$$\mathbf{K}'_{Po} = 2\mathbf{E}^{\top}(\eta_{de}, \epsilon_{de}) \mathbf{R}_e \mathbf{K}_{Po} \mathbf{R}_e^{\top}$$

and $\mathbf{E}(\eta_{de}, \epsilon_{de}) = \eta_{de} \mathbf{I} - \mathbf{S}(\epsilon_{de})$. The above equations show that a finite orientation displacement $\epsilon_{de} = \mathbf{R}_e^{\top} \epsilon_{de}^e$ produces an elastic wrench $\mathbf{h}_{\Delta o} = (\mathbf{0}^{\top} \mathbf{m}_{\Delta o}^{\top})^{\top}$ which is equivalent to a pure moment.

Hence, the total elastic wrench in the presence of a finite position and orientation displacement of the desired frame Σ_d with respect to the end-effector frame Σ_e can be defined in the base frame as

$$\mathbf{h}_{\Delta} = \mathbf{h}_{\Delta t} + \mathbf{h}_{\Delta o}. \quad (7.13)$$

where $\mathbf{h}_{\Delta t}$ and $\mathbf{h}_{\Delta o}$ are computed according to (7.10) and (7.12), respectively.

Using (7.13) for the computation of the elastic wrench in the case of an infinitesimal twist displacement $\delta \mathbf{x}_{de}^e$ near the equilibrium, and discarding the high-order infinitesimal terms, yields the linear mapping

$$\mathbf{h}_e^e = \mathbf{K}_P \delta \mathbf{x}_{de}^e = \begin{pmatrix} \mathbf{K}_{Pt} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{Po} \end{pmatrix} \delta \mathbf{x}_{de}^e. \quad (7.14)$$

Therefore, \mathbf{K}_P represents the stiffness matrix of an ideal spring with respect to a frame Σ_e (coinciding with Σ_d at equilibrium) with the origin at the center of stiffness. Moreover, it can be shown, using definition (7.13), that the physical/geometrical meaning of the principal stiffnesses and of the principal axes for the matrices \mathbf{K}_{Pt} and \mathbf{K}_{Po} are preserved also in the case of large displacements.

The above results imply that the active stiffness matrix \mathbf{K}_P can be set in a geometrically consistent way with respect to the task at hand.

Notice that geometrical consistency can also be ensured with different definitions of the orientation error in the potential orientation energy (7.11), for example, any error based on the angle/axis representation of \mathbf{R}_e^d can be adopted (the unit quaternion belongs to this category), or, more generally, homogeneous matrices or exponential coordinates (for the case of both position and orientation errors). Also, the XYZ Euler angles extracted from the matrix \mathbf{R}_e^d could be used; however, in this case, it can be shown that the principal axes of rotational stiffness cannot be set arbitrarily but must coincide with those of the end-effector frame.

Compliance control with a geometrically consistent active stiffness can be defined using the control law

$$\mathbf{h}_e = \mathbf{h}_{\Delta} - \mathbf{K}_D \mathbf{v}_e + \eta(\mathbf{q}),$$

with \mathbf{h}_{Δ} in (7.13). The asymptotic stability about the equilibrium in the case $\mathbf{h}_e = \mathbf{0}$ can be proven using the Lyapunov function

$$V = \frac{1}{2} \mathbf{v}_e^{\top} \mathbf{A}(\mathbf{q}) \mathbf{v}_e + V_t + V_0,$$

with V_t and V_0 given in (7.9) and (7.11), respectively, whose time derivative along the trajectories of the closed-loop system, in case the frame Σ_d is motionless, has the same expression as in (7.4). When $\mathbf{h}_e \neq \mathbf{0}$, a different asymptotically stable equilibrium can be found, corresponding to a nonnull displacement of the desired frame Σ_d with respect to the end-effector frame Σ_e . The new equilibrium is the solution of the equation $\mathbf{h}_{\Delta} = \mathbf{h}_e$.

Stiffness control allows to keep the interaction force and moment limited at the expense of the end-effector position and orientation error, with a proper choice of the stiffness matrix, without the need of a force/torque sensor. However, in the presence of disturbances (e.g., joint friction) which can be modeled as an equivalent end-effector wrench, the adoption of low values for the active stiffness may produce large deviations with respect to the desired end-effector position and orientation, also in the absence of interaction with the environment.

7.2.2 Impedance Control

Stiffness control is designed to achieve a desired *static* behavior of the interaction. In fact, the dynamics of the controlled system depends on that of the robot manipulator, which is nonlinear and coupled. A more demanding objective may be that of achieving a desired *dynamic* behavior for the end-effector, e.g., that of a second-order mechanical system with six degrees of freedom, characterized by a given mass, damping, and stiffness, known as mechanical *impedance*.

The starting point to pursue this goal may be the acceleration-resolved approach used for motion control, which is aimed at decoupling and linearizing the nonlinear robot dynamics at the acceleration level via an inverse dynamics control law. In the presence of interaction with the environment, the control law

$$\mathbf{h}_e = \mathbf{A}(\mathbf{q}) \boldsymbol{\alpha} + \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}} + \mathbf{h}_e \quad (7.15)$$

cast into the dynamic model (7.2) results in

$$\dot{\mathbf{v}}_e = \boldsymbol{\alpha}, \quad (7.16)$$

where α is a properly designed control input with the meaning of an acceleration referred to the base frame. Considering the identity $\dot{\mathbf{v}}_e = \bar{\mathbf{R}}_e^\top \dot{\mathbf{v}}_e^e + \dot{\bar{\mathbf{R}}}_e^\top \mathbf{v}_e^e$, with

$$\bar{\mathbf{R}}_e = \begin{pmatrix} \mathbf{R}_e & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_e \end{pmatrix},$$

the choice

$$\alpha = \bar{\mathbf{R}}_e^\top \alpha^e + \dot{\bar{\mathbf{R}}}_e^\top \mathbf{v}_e^e \quad (7.17)$$

gives

$$\dot{\mathbf{v}}_e^e = \alpha^e, \quad (7.18)$$

where the control input α^e has the meaning of an acceleration referred to the end-effector frame Σ_e . Hence, setting

$$\alpha^e = \mathbf{K}_M^{-1}(\dot{\mathbf{v}}_d^e + \mathbf{K}_D \Delta \mathbf{v}_{de}^e + \mathbf{h}_\Delta^e - \mathbf{h}_e^e), \quad (7.19)$$

the following expression can be found for the closed-loop system

$$\mathbf{K}_M \Delta \dot{\mathbf{v}}_{de}^e + \mathbf{K}_D \Delta \mathbf{v}_{de}^e + \mathbf{h}_\Delta^e = \mathbf{h}_e^e, \quad (7.20)$$

where \mathbf{K}_M and \mathbf{K}_D are 6×6 symmetric and positive-definite matrices, $\Delta \dot{\mathbf{v}}_{de}^e = \dot{\mathbf{v}}_d^e - \dot{\mathbf{v}}_e^e$, $\Delta \mathbf{v}_{de}^e = \mathbf{v}_d^e - \mathbf{v}_e^e$, $\dot{\mathbf{v}}_d^e$ and \mathbf{v}_d^e are, respectively, the acceleration and the velocity of a desired frame Σ_d and \mathbf{h}_Δ^e is the elastic wrench (7.13); all the quantities are referred to the end-effector frame Σ_e .

The above equation describing the dynamic behavior of the controlled end-effector can be interpreted as a generalized mechanical impedance. The asymptotic stability of the equilibrium in the case $\mathbf{h}_e = \mathbf{0}$ can be proven by considering the Lyapunov function

$$V = \frac{1}{2} \Delta \mathbf{v}_{de}^{e\top} \mathbf{K}_M \Delta \mathbf{v}_{de}^e + V_t + V_o, \quad (7.21)$$

where V_t and V_o are defined in (7.9) and (7.11), respectively, and whose time derivative along the trajectories of system (7.20) is the negative semidefinite function

$$\dot{V} = -\Delta \mathbf{v}_{de}^{e\top} \mathbf{K}_D \Delta \mathbf{v}_{de}^e.$$

When $\mathbf{h}_e \neq \mathbf{0}$, a different asymptotically stable equilibrium can be found, corresponding to a nonnull displacement of the desired frame Σ_d with respect to the end-effector frame Σ_e . The new equilibrium is the solution of the equation $\mathbf{h}_\Delta^e = \mathbf{h}^e$.

In case Σ_d is constant, (7.20) has the meaning of a true 6-DOF mechanical impedance if \mathbf{K}_M is chosen as

$$\mathbf{K}_M = \begin{pmatrix} m\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} \end{pmatrix},$$

where m is a mass and \mathbf{M} is a 3×3 inertia tensor, and \mathbf{K}_D is chosen as a block-diagonal matrix with 3×3 blocks. The physically equivalent system is a body of mass m with inertia tensor \mathbf{M} with respect to a frame Σ_e attached to the body, subject to an external wrench \mathbf{h}^e . This body is connected to a virtual body attached to frame Σ_d through a 6-DOF ideal spring with stiffness matrix \mathbf{K}_P and is subject to viscous forces and moments with damping \mathbf{K}_D . The function V in (7.21) represents the total energy of the body: the sum of the kinetic and potential elastic energy.

A block diagram of the resulting impedance control is sketched in Fig. 7.1. The impedance control computes the acceleration input as in (7.17) and (7.19) on the basis of the position and orientation feedback as well as the force and moment measurements. Then, the inverse dynamics control law computes the torques for the joint actuators $\boldsymbol{\tau} = \mathbf{J}^\top \mathbf{h}_c$ with \mathbf{h}_c in (7.15). This control scheme, in the absence of interaction, guarantees that the end-effector frame Σ_e asymptotically follows the desired frame Σ_d . In the presence of contact with the environment, a compliant dynamic behavior is imposed on the end-effector, according to the impedance (7.20), and the contact wrench is bounded at the expense of a finite position and orientation displacement between Σ_d and Σ_e . Differently from stiffness control, a force/torque sensor is required for the measurement of the contact force and moment.

Implementation Issues

The selection of good impedance parameters ensuring a satisfactory behavior is not an easy task. In fact, the dynamics of the closed-loop system is different in free space and during interaction. The control objectives are different as well, since motion tracking and disturbance

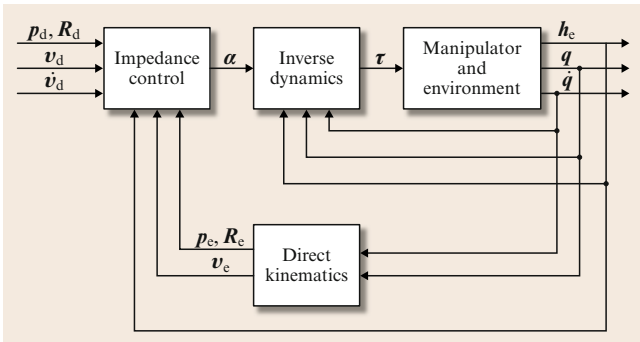


Fig. 7.1 Impedance control

rejection must be ensured in free space, while, during the interaction, the main goal is achieving a suitable compliant dynamic behavior for the end-effector. Notice also that the dynamics of the controlled system during the interaction depends on the dynamics of the environment.

To gain insight into these problems, assume that the interaction of the end-effector with the environment can be approximated by that derived from an ideal 6-DOF spring connecting end-effector frame Σ_e to the environment frame Σ_o . Therefore, according to (7.8), the elastic wrench exerted by the end-effector on the environment, in the presence of an infinitesimal twist displacement of Σ_e with respect to Σ_o , can be computed as

$$\mathbf{h}_e^e = \mathbf{K} \delta \mathbf{x}_{e0}^e, \quad (7.22)$$

where Σ_e and Σ_o coincide at equilibrium and \mathbf{K} is a stiffness matrix. The above model holds only in the presence of interaction, while the contact wrench is null when the end-effector moves in free space.

The disturbances acting on the robot manipulator and the unmodeled dynamics (joint friction, modeling errors, etc.) may be taken into account by introducing an additive term on the right-hand side of the dynamic model of the robot manipulator (7.2), corresponding to an equivalent disturbance wrench acting on the end-effector. This term produces an additive acceleration disturbance $\boldsymbol{\gamma}^e$ on the right-hand side of (7.18). Therefore, using the control law (7.19), the following closed-loop impedance equation can be found

$$\mathbf{K}_M \Delta \dot{\mathbf{v}}_d^e + \mathbf{K}_D \Delta \mathbf{v}_d^e + \mathbf{h}_\Delta^e = \mathbf{h}_e^e + \mathbf{K}_M \boldsymbol{\gamma}^e. \quad (7.23)$$

The tuning procedure for the impedance parameters can be set up starting from the linearized model that can be computed from (7.23) in the case of infinitesimal displacements, i. e.:

$$\begin{aligned} \mathbf{K}_M \delta \ddot{\mathbf{x}}_{de}^e + \mathbf{K}_D \delta \dot{\mathbf{x}}_{de}^e + (\mathbf{K}_P + \mathbf{K}) \delta \mathbf{x}_{de}^e \\ = \mathbf{K} \delta \mathbf{x}_{do}^e + \mathbf{K}_M \boldsymbol{\gamma}^e, \end{aligned} \quad (7.24)$$

where (7.22) and the equality $\delta \mathbf{x}_{e0}^e = -\delta \mathbf{x}_{de}^e + \delta \mathbf{x}_{do}^e$ have been used. The above equation is valid both for constrained ($\mathbf{K} \neq \mathbf{0}$) and for free motion ($\mathbf{K} = \mathbf{0}$).

It is evident that suitable dynamics of the position and orientation errors can be set by suitably choosing the matrix gains \mathbf{K}_M , \mathbf{K}_D , and \mathbf{K}_P . This task is easier under the hypothesis that all the matrices are diagonal, resulting in a decoupled behavior for the six components of the infinitesimal twist displacement. In this case, the transient behavior of each component can be set, e.g., by assigning the natural frequency and damping ratio with

the relations

$$\omega_n = \sqrt{\frac{k_P + k}{k_M}}, \quad \zeta = \frac{1}{2} \frac{k_D}{\sqrt{k_M(k_P + k)}}.$$

Hence, if the gains are chosen so that a given natural frequency and damping ratio are ensured during the interaction (i. e., for $k \neq 0$), a smaller natural frequency with a higher damping ratio will be obtained when the end-effector moves in free space (i. e., for $k = 0$). As for the steady-state performance, the end-effector error for the generic component is

$$\delta x_{de} = \frac{k}{(k_P + k)} \delta x_{do} + \frac{k_M}{k_P + k} \boldsymbol{\gamma}$$

and the corresponding interaction force is

$$h = \frac{k_P k}{k_P + k} \delta x_{do} - \frac{k_M k}{k_P + k} \boldsymbol{\gamma}.$$

The above relations show that, during interaction, the contact force can be made small at the expense of a large position error in steady state, as long as the active stiffness k_P is set low with respect to the stiffness of the environment k , and vice versa. However, both the contact force and the position error also depend on the external disturbance $\boldsymbol{\gamma}$; in particular, the lower k_P , the higher the influence of $\boldsymbol{\gamma}$ on both δx_{de} and h . Moreover, a low active stiffness k_P may result in a large position error also in the absence of interaction (i. e., when $k = 0$).

Admittance Control

A solution to this drawback can be devised by separating motion control from impedance control as follows. The motion control action is purposefully made stiff so as to enhance disturbance rejection but, rather than ensuring tracking of the desired end-effector position and orientation, it ensures tracking of a reference position and orientation resulting from the impedance control action. In other words, the desired position and orientation, together with the measured contact wrench, are input to the impedance equation which, via a suitable integration, generates the position and orientation to be used as a reference for the motion control.

To implement this solution, it is worth introducing a reference frame other than the desired frame Σ_d . This frame is referred to as the *compliant frame* Σ_c , and is specified by the quantities \mathbf{p}_c , \mathbf{R}_c , \mathbf{v}_c , and $\dot{\mathbf{v}}_c$ that are computed from \mathbf{p}_d , \mathbf{R}_d , \mathbf{v}_d , and $\dot{\mathbf{v}}_d$ and the measured wrench \mathbf{h}_c , by integrating the equation

$$\mathbf{K}_M \Delta \dot{\mathbf{v}}_{dc}^c + \mathbf{K}_D \Delta \mathbf{v}_{dc}^c + \mathbf{h}_\Delta^c = \mathbf{h}^c, \quad (7.25)$$

where \mathbf{h}_Δ^c is the elastic wrench in the presence of a finite displacement between the desired frame Σ_d and the

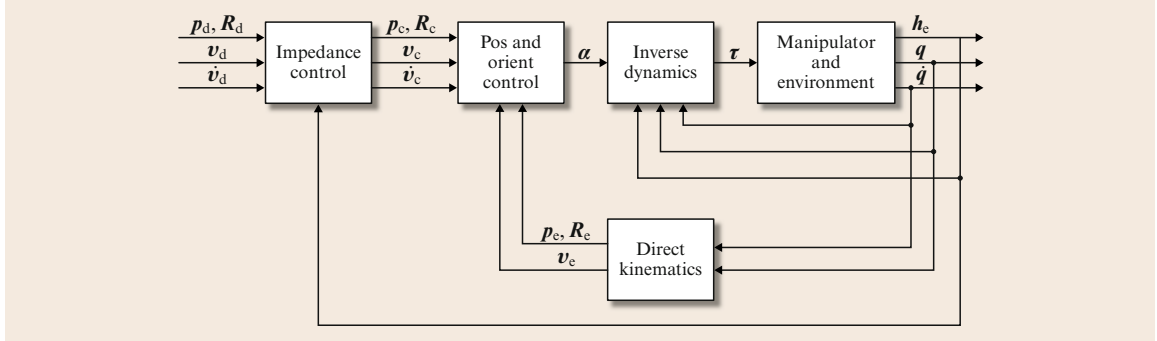


Fig. 7.2 Impedance control with inner motion control loop (admittance control)

compliant frame Σ_c . Then, a motion control strategy, based on inverse dynamics, is designed so that the end-effector frame Σ_e is taken to coincide with the compliant frame Σ_c . To guarantee the stability of the overall system, the bandwidth of the motion controller should be higher than the bandwidth of the impedance controller.

A block diagram of the resulting scheme is sketched in Fig. 7.2. It is evident that, in the absence of interaction, the compliant frame Σ_c coincides with the desired frame Σ_d and the dynamics of the position and orientation error, as well as the disturbance rejection capabilities, depend only on the gains of the inner motion control loop. On the other hand, the dynamic behavior in the presence of interaction is imposed by the impedance gains (7.25).

The control scheme of Fig. 7.2 is also known as *admittance control* because, in (7.25), the measured force (the input) is used to compute the motion of the compliant frame (the output), given the motion of the desired frame; a mapping with a force as input and a position or velocity as output corresponds to a mechanical *admittance*. Vice versa, (7.20), mapping the end-effector displacement (the input) from the desired motion trajectory into the contact wrench (the output), has the meaning of a mechanical *impedance*.

Simplified Schemes

The inverse dynamics control is model based and requires modification of current industrial robot controllers, which are usually equipped with independent PI joint velocity controllers with very high bandwidth. These controllers are able to decouple the robot dynamics to a large extent, especially in the case of slow motion, and to mitigate the effects of external forces on the manipulator motion if the environment is sufficiently compliant. Hence, the closed-loop dynamics of the controlled robot can be approximated by

$$\dot{q} = \dot{q}_r$$

in joint space, or equivalently

$$\dot{v}_e = v_r \quad (7.26)$$

in the operational space, where \dot{q}_r and v_r are the control signals for the inner velocity motion loop generated by a suitably designed outer control loop. These control signals are related by

$$\dot{q}_r = J^{-1}(q)v_r.$$

The velocity v_r , corresponding to a *velocity-resolved control*, can be computed as

$$v_r^e = v_{dc}^e + K_D^{-1}(h_{\Delta}^e - h_c^e),$$

where the control input has been referred to the end-effector frame, K_D is a 6×6 positive-definite matrix and h_{Δ} is the elastic wrench (7.13) with stiffness matrix K_P . The resulting closed-loop equation is

$$K_D \Delta v_{dc}^e + h_{\Delta}^e = h_c^e$$

corresponding to a compliant behavior of the end-effector characterized by a damping K_D and a stiffness K_P . In the case $K_P = 0$, the resulting scheme is known as *damping control*.

Alternatively, an admittance-type control scheme can be adopted, where the motion of a compliant frame Σ_c can be computed as the solution of the differential equation

$$K_D \Delta v_{dc}^c + h_{\Delta}^c = h_c^c$$

in terms of the position p_c , orientation R_c , and velocity twist v_c , where the inputs are the motion variables of the desired frame Σ_d and the contact wrench h_c^c . The motion variables of Σ_c are then input to an inner position and velocity controller. In the case $K_D = 0$, the resulting scheme is known as *compliance control*.

7.3 Interaction Tasks

Indirect force control does not require explicit knowledge of the environment, although to achieve a satisfactory dynamic behavior the control parameters have to be tuned for a particular task. On the other hand, a model of the interaction task is required for the synthesis of direct force control algorithms.

An interaction task is characterized by complex contact situations between the manipulator and the environment. To guarantee proper task execution, it is necessary to have an analytical description of the interaction force and moment, which is very demanding from a modeling viewpoint.

A real contact situation is a naturally distributed phenomenon in which the local characteristics of the contact surfaces as well as the global dynamics of the manipulator and environment are involved. In detail:

- the environment imposes kinematic constraints on the end-effector motion, due to one or more contacts of different type, and a reaction wrench arises when the end-effector tends to violate the constraints (e.g., the case of a robot sliding a rigid tool on a frictionless rigid surface);
- the end-effector, while being subject to kinematic constraints, may also exert a dynamic wrench on the environment, in the presence of environment dynamics (e.g., the case of a robot turning a crank, when the crank dynamics is relevant, or a robot pushing against a compliant surface);
- the contact wrench may depend on the structural compliance of the robot, due to the finite stiffness of the joints and links of the manipulator, as well as of the wrist force/torque sensor or of the tool (e.g., an end-effector mounted on an [RCC](#) device);
- local deformation of the contact surfaces may occur during the interaction, producing distributed contact areas (e.g., the case of a soft contact surface of the tool or of the environment);
- static and dynamic friction may occur in the case of non ideally smooth contact surfaces.

The design of the interaction control and the performance analysis are usually carried out under simplifying assumptions. The following two cases are considered:

1. the robot and the environment are perfectly rigid and purely kinematics constraints are imposed by the environment,
2. the robot is perfectly rigid, all the compliance in the system is localized in the environment, and the

contact wrench is approximated by a linear elastic model.

In both cases, frictionless contact is assumed. It is obvious that these situations are only ideal. However, the robustness of the control should be able to cope with situations where some of the ideal assumptions are relaxed. In that case the control laws may be adapted to deal with nonideal characteristics.

7.3.1 Rigid Environment

The kinematic constraints imposed by the environment can be represented by a set of equations that the variables describing the end-effector position and orientation must satisfy; since these variables depend on the joint variables through the direct kinematic equations, the constraint equations can also be expressed in the joint space as

$$\phi(q) = 0. \quad (7.27)$$

The vector ϕ is an $m \times 1$ function, with $m < n$, where n is the number of joints of the manipulator, assumed to be nonredundant; without loss of generality, the case $n = 6$ is considered. Constraints of the form (7.27), involving only the generalized coordinates of the system, are known as holonomic constraints. The case of time-varying constraints of the form $\phi(q, t) = 0$ is not considered here but can be analyzed in a similar way. Moreover, only bilateral constraints expressed by equalities of the form (7.27) are of concern; this means that the end-effector always keeps contact with the environment. The analysis presented here is known as *kinetostatic analysis*.

It is assumed that the vector (7.27) is twice differentiable and that its m components are linearly independent at least locally in a neighborhood of the operating point. Hence, differentiation of (7.27) yields

$$J_\phi(q)\dot{q} = 0, \quad (7.28)$$

where $J_\phi(q) = \partial\phi/\partial q$ is the $m \times 6$ Jacobian of $\phi(q)$, known as the *constraint Jacobian*. By virtue of the above assumption, $J_\phi(q)$ is of rank m at least locally in a neighborhood of the operating point.

In the absence of friction, the generalized interaction forces are represented by a reaction wrench that tends to violate the constraints. This end-effector wrench produces reaction torques at the joints that can be computed

using the principle of virtual work as

$$\tau_e = J_\phi^\top(q)\lambda,$$

where λ is an $m \times 1$ vector of Lagrange multipliers. The end-effector wrench corresponding to τ_e can be computed as

$$h_e = J^{-\top}(q)\tau_e = S_f(q)\lambda, \quad (7.29)$$

where

$$S_f = J^{-\top}(q)J_\phi^\top(q). \quad (7.30)$$

From (7.29) it follows that h_e belongs to the m -dimensional vector space spanned by the columns of the $6 \times m$ matrix S_f . The inverse of the linear transformation (7.29) is computed as

$$\lambda = S_f^\dagger(q)h_e, \quad (7.31)$$

where S_f^\dagger denotes a weighted pseudoinverse of the matrix S_f , i. e.,

$$S_f^\dagger = (S_f^\top W S_f)^{-1} S_f^\top W \quad (7.32)$$

where W is a suitable weighting matrix.

Notice that, while the range space of the matrix S_f in (7.30) is uniquely defined by the geometry of the contact, the matrix S_f itself is not unique; also, the constraint equations (7.27), the corresponding Jacobian J_ϕ as well as the pseudoinverse S_f^\dagger and the vector λ are not uniquely defined.

In general, the physical units of measure of the elements of λ are not homogeneous and the columns of the matrix S_f , as well as of the matrix S_f^\dagger , do not necessarily represent homogeneous entities. This may produce invariance problems in the transformation (7.31) if h_e represents a measured wrench that is subject to disturbances and, as a result, may have components outside the range space of S_f . If a physical unit or a reference frame is changed, the matrix S_f undergoes a transformation; however, the result of (7.31) with the transformed pseudoinverse in general depends on the adopted physical units or on the reference frame. The reason is that the pseudoinverse is the weighted least-squares solution of a minimization problem based on the norm of the vector $h_e - S_f(q)\lambda$, and the invariance can be guaranteed only if a physically consistent norm of this vector is used. In the ideal case that h_e is in the range space of S_f , there is a unique solution for λ in (7.31), regardless of the weighting matrix, and hence the invariance problem does not appear.

A possible solution consists of choosing S_f so that its columns represent linearly independent wrenches. This

implies that (7.29) gives h_e as a linear combination of wrenches and λ is a dimensionless vector. A physically consistent norm on the wrench space can be defined based on the quadratic form $h_e^\top K^{-1} h_e$, which has the meaning of an elastic energy if K is a positive-definite matrix corresponding to a stiffness. Hence, the choice $W = K^{-1}$ can be made for the weighting matrix of the pseudoinverse.

Notice that, for a given S_f , the constraint Jacobian can be computed from (7.30) as $J_\phi(q) = S_f^\top J(q)$; moreover, the constraint equations can be derived by integrating (7.28).

Using (7.1) and (7.30), the equality (7.28) can be rewritten in the form

$$J_\phi(q)J^{-1}(q)J(q)\dot{q} = S_f^\top v_e = 0, \quad (7.33)$$

which, by virtue of (7.29), is equivalent to

$$h_e^\top v_e = 0. \quad (7.34)$$

Equation (7.34) represents the kinetostatic relationship, known as *reciprocity*, between the ideal reaction wrench h_e (belonging to the so-called *force-controlled* subspace) and the end-effector twist that obeys the constraints (belonging to the so-called *velocity-controlled* subspace). The concept of reciprocity, expressing the physical fact that, in the hypothesis of rigid and frictionless contact, the wrench does not cause any work against the twist, is often confused with the concept of *orthogonality*, which makes no sense in this case because twists and wrenches belong to different spaces.

Equations (7.33) and (7.34) imply that the velocity-controlled subspace is the reciprocal complement of the m -dimensional force-controlled subspace, identified by the range of matrix S_f . Hence, the dimension of the velocity-controlled subspace is $6 - m$ and a $6 \times (6 - m)$ matrix S_v can be defined, whose columns span the velocity-controlled subspace, i. e.,

$$v_e = S_v(q)v, \quad (7.35)$$

where v is a suitable $(6 - m) \times 1$ vector. From (7.33) and (7.35) the following equality holds

$$S_f^\top(q)S_v(q) = 0; \quad (7.36)$$

moreover, the inverse of the linear transformation (7.35) can be computed as

$$v = S_v^\dagger(q)v_e, \quad (7.37)$$

where S_v^\dagger denotes a suitable weighted pseudoinverse of the matrix S_v , computed as in (7.32).

Notice that, as for the case of S_f , although the range space of the matrix S_v is uniquely defined, the choice of the matrix S_v itself is not unique. Moreover, the columns of S_v are not necessarily twists and the scalar v may have nonhomogeneous physical dimensions. However, in order to avoid invariance problems analogous to that considered for the case of S_f , it is convenient to select the columns of S_v as twists so that the vector v is dimensionless; moreover, the weighting matrix used to compute the pseudoinverse in (7.37) can be set as $W = M$, being M a 6×6 inertia matrix; this corresponds to defining a norm in the space of twists based on the kinetic energy. It is worth observing that the transformation matrices of twists and wrenches, corresponding to a change of reference frame, are different; however, if twists are defined with angular velocity on top and translational velocity on bottom, then their transformation matrix is the same as for wrenches.

The matrix S_v may also have an interpretation in terms of Jacobians, as for S_f in (7.30). Due to the presence of m independent holonomic constraints (7.27), the configuration of the robot in contact with the environment can be described in terms of a $(6 - m) \times 1$ vector r of independent variables. From the implicit function theorem, this vector can be defined as

$$r = \psi(q), \quad (7.38)$$

where $\psi(q)$ is any $(6 - m) \times 1$ twice-differentiable vector function such that the m components of $\phi(q)$ and the $n - m$ components of $\psi(q)$ are linearly independent at least locally in a neighborhood of the operating point. This means that the mapping (7.38), together with the constraint (7.27), is locally invertible, with inverse defined as

$$q = \rho(r), \quad (7.39)$$

where $\rho(r)$ is a 6×1 twice-differentiable vector function. Equation (7.39) explicitly provides all the joint vectors which satisfy the constraint (7.27). Moreover, the joint velocity vectors that satisfy (7.28) can be computed as

$$\dot{q} = J_\rho(r)\dot{r},$$

where $J_\rho(r) = \partial \rho / \partial r$ is a $6 \times (6 - m)$ full-rank Jacobian matrix. Therefore, the following equality holds

$$J_\phi(q)J_\rho(r) = 0,$$

which can be interpreted as a reciprocity condition between the subspace of the reaction torques spanned by the columns of the matrix J_ϕ^\top and the subspace of the

constrained joint velocities spanned by the columns of the matrix J_ρ .

Rewriting the above equation as

$$J_\phi(q)J(q)^{-1}J(q)J_\rho(r) = 0,$$

and taking into account (7.30) and (7.36), the matrix S_v can be expressed as

$$S_v = J(q)J_\rho(r), \quad (7.40)$$

which, by virtue of (7.38) and (7.39), it can be equivalently expressed as a function of either q or r .

The matrices S_f and S_v , and their pseudoinverse S_f^\dagger and S_v^\dagger are known as selection matrices. They play a fundamental role for the task specification, i.e., the specification of the desired end-effector motion and interaction forces and moments, as well as for the control synthesis.

7.3.2 Compliant Environment

In many applications, the interaction wrench between the end-effector and a compliant environment can be approximated by an ideal elastic model of the form (7.22). However, since the stiffness matrix K is positive definite, this model describes a fully constrained case, when the environment deformation coincides with the infinitesimal twist displacement of the end-effector. In general, however, the end-effector motion is only partially constrained by the environment and this situation can be modeled by introducing a suitable positive-semidefinite stiffness matrix.

The stiffness matrix describing the partially constrained interaction between the end-effector and the environment can be computed by modeling the environment as a couple of rigid bodies, S and O, connected through an ideal 6-DOF spring of compliance $C = K^{-1}$. Body S is attached to a frame Σ_s and is in contact with the end-effector; body O is attached to a frame Σ_o which, at equilibrium, coincides with frame Σ_s . The environment deformation about the equilibrium, in the presence of a wrench h_s , is represented by the infinitesimal twist displacement δx_{so} between frames Σ_s and Σ_o , which can be computed as

$$\delta x_{so} = C h_s. \quad (7.41)$$

All the quantities hereafter are referred to frame Σ_s but the superscript s is omitted for brevity.

For the considered contact situation, the end-effector twist does not completely belong to the ideal velocity subspace, corresponding to a rigid environment, because

the environment can deform. Therefore, the infinitesimal twist displacement of the end-effector frame Σ_e with respect to Σ_o can be decomposed as

$$\delta \mathbf{x}_{eo} = \delta \mathbf{x}_v + \delta \mathbf{x}_f, \quad (7.42)$$

where $\delta \mathbf{x}_v$ is the end-effector infinitesimal twist displacement in the velocity-controlled subspace, defined as the $6-m$ reciprocal complement of the force-controlled subspace, while $\delta \mathbf{x}_f$ is the end-effector infinitesimal twist displacement corresponding to the environment deformation. Hence:

$$\delta \mathbf{x}_v = \mathbf{P}_v \delta \mathbf{x}_{eo} \quad (7.43)$$

$$\delta \mathbf{x}_f = (\mathbf{I} - \mathbf{P}_v) \delta \mathbf{x}_{eo} = (\mathbf{I} - \mathbf{P}_v) \delta \mathbf{x}_{so}, \quad (7.44)$$

where $\mathbf{P}_v = \mathbf{S}_v \mathbf{S}_v^\dagger$ and \mathbf{S}_v and \mathbf{S}_v^\dagger are defined as in the rigid environment case. The matrix \mathbf{P}_v is a projection matrix that filters out all the end-effector twists (and infinitesimal twist displacements) that are not in the range space of \mathbf{S}_v , while $\mathbf{I} - \mathbf{P}_v$ is a projection matrix that filters out all the end-effector twists (and infinitesimal twist displacements) that are in the range space of \mathbf{S}_v . The twists $\mathbf{P}_v \mathbf{v}$ are denoted as *twists of freedom* while the twists $(\mathbf{I} - \mathbf{P}_v) \mathbf{v}$ are denoted as *twists of constraint*.

In the hypothesis of frictionless contact, the interaction wrench between the end-effector and the environment is restricted to a force-controlled subspace defined by the m -dimensional range space of a matrix \mathbf{S}_f , as for the rigid environment case, i.e.,

$$\mathbf{h}_e = \mathbf{S}_f \boldsymbol{\lambda} = \mathbf{h}_s, \quad (7.45)$$

where $\boldsymbol{\lambda}$ is an $m \times 1$ dimensionless vector. Premultiplying both sides of (7.42) by \mathbf{S}_f^\top and using (7.41), (7.43), (7.44), and (7.45) yields

$$\mathbf{S}_f^\top \delta \mathbf{x}_{eo} = \mathbf{S}_f^\top \mathbf{C} \mathbf{S}_f \boldsymbol{\lambda},$$

where the identity $\mathbf{S}_f^\top \mathbf{P}_v = \mathbf{0}$ has been exploited. Therefore, the following elastic model can be found:

$$\mathbf{h}_e = \mathbf{S}_f \boldsymbol{\lambda} = \mathbf{K}' \delta \mathbf{x}_{eo}, \quad (7.46)$$

where $\mathbf{K}' = \mathbf{S}_f (\mathbf{S}_f^\top \mathbf{C} \mathbf{S}_f)^{-1} \mathbf{S}_f^\top$ is the positive-semidefinite stiffness matrix corresponding to the partially constrained interaction.

If the compliance matrix \mathbf{C} is adopted as a weighting matrix for the computation of \mathbf{S}_f^\dagger , then \mathbf{K}' can be expressed as

$$\mathbf{K}' = \mathbf{P}_f \mathbf{K}, \quad (7.47)$$

where $\mathbf{P}_f = \mathbf{S}_f \mathbf{S}_f^\dagger$ is a projection matrix that filters out all the end-effector wrenches that are not in the range space of \mathbf{S}_f .

The compliance matrix for the partially constrained interaction cannot be computed as the inverse of \mathbf{K}' , since this matrix is of rank $m < 6$. However, using (7.44), (7.41), and (7.45), the following equality can be found

$$\delta \mathbf{x}_f = \mathbf{C}' \mathbf{h}_e,$$

where the matrix

$$\mathbf{C}' = (\mathbf{I} - \mathbf{P}_v) \mathbf{C}, \quad (7.48)$$

of rank $6-m$, is positive semidefinite. If the stiffness matrix \mathbf{K} is adopted as a weighting matrix for the computation of \mathbf{S}_v^\dagger , then the matrix \mathbf{C}' has the noticeable expression $\mathbf{C}' = \mathbf{C} - \mathbf{S}_v (\mathbf{S}_v^\top \mathbf{K} \mathbf{S}_v)^{-1} \mathbf{S}_v^\top$, showing that \mathbf{C}' is symmetric.

7.3.3 Task Specification

An interaction task can be assigned in terms of a desired end-effector wrench \mathbf{h}_d and twist \mathbf{v}_d . In order to be consistent with the constraints, these vectors must lie in the force- and velocity-controlled subspaces, respectively. This can be guaranteed by specifying vectors $\boldsymbol{\lambda}_d$ and \mathbf{v}_d and computing \mathbf{h}_d and \mathbf{v}_d as

$$\mathbf{h}_d = \mathbf{S}_f \boldsymbol{\lambda}_d, \quad \mathbf{v}_d = \mathbf{S}_v \mathbf{v}_d,$$

where \mathbf{S}_f and \mathbf{S}_v have to be suitably defined on the basis of the geometry of the task, and so that invariance with respect to the choice of the reference frame and change of physical units is guaranteed.

Many robotic tasks have a set of orthogonal reference frames in which the task specification is very easy and intuitive. Such frames are called *task frames* or *compliance frames*. An interaction task can be specified by assigning a desired force/torque or a desired linear/angular velocity along/about each of the frame axes. The desired quantities are known as *artificial constraints* because they are imposed by the controller; these constraints, in the case of rigid contact, are complementary to those imposed by the environment, known as *natural constraints*.

Some examples of task frame definition and task specification are given below.

Peg-in-Hole

The goal of this task is to push the peg into the hole while avoiding wedging and jamming. The peg has two degrees of motion freedom, hence the dimension of the velocity-controlled subspace is $6-m=2$, while the dimension of the force-controlled subspace is $m=4$. The task frame can be chosen as shown in Fig. 7.3 and the

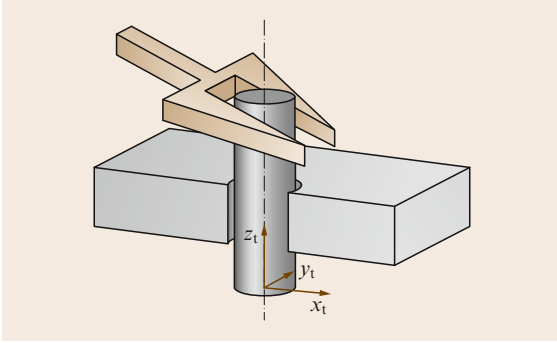


Fig. 7.3 Insertion of a cylindrical peg into a hole

task can be achieved by assigning the following desired forces and torques:

- zero forces along the x_t and y_t axes
- zero torques about the x_t and y_t axes

and the desired velocities

- a nonzero linear velocity along the z_t -axis
- an arbitrary angular velocity about the z_t -axis

The task continues until a *large* reaction force in the z_t direction is measured, indicating that the peg has hit the bottom of the hole, not represented in the figure. Hence, the matrices S_f and S_v can be chosen as

$$S_f = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad S_v = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix},$$

where the columns of S_f have the dimensions of wrenches and those of S_v have the dimensions of twists, defined in the task frame, and they transform accordingly when changing the reference frame. The task frame can be chosen either attached to the end-effector or to the environment.

Turning a Crank

The goal of this task is turning a crank with an idle handle. The handle has two degrees of motion freedom, corresponding to the rotation about the z_t -axis and to the rotation about the rotation axis of the crank. Hence the dimension of the velocity-controlled subspace is $6 - m = 3$ while the dimension of the force-controlled subspace is $m = 4$. The task frame can be assumed as in the Fig. 7.4, attached to the crank. The task can be

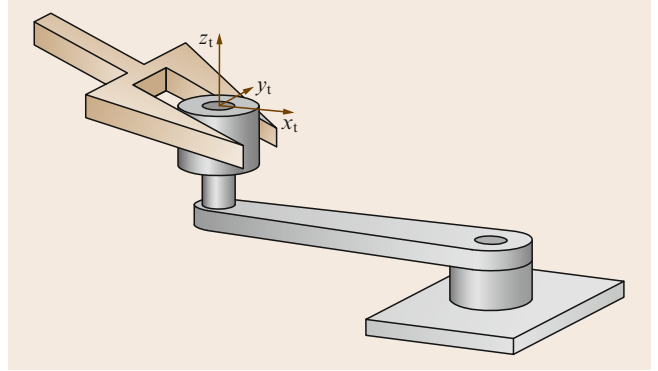


Fig. 7.4 Turning a crank with an idle handle

achieved by assigning the following desired forces and torques:

- zero forces along the x_t and z_t axes
- zero torques about the x_t and y_t axes

and the following desired velocities

- a nonzero linear velocity along the y_t -axis
- an arbitrary angular velocity about the z_t -axis

Hence, the matrices S_f and S_v can be chosen as

$$S_f = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad S_v = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix},$$

referred to the task frame. In this case, the task frame is fixed with respect to the crank, but in motion with respect both the end-effector frame (fixed to the handle) and to the base frame of the robot. Hence, the matrices S_f and S_v are time variant when referred either to the end-effector frame or to the base frame.

Sliding a Block on a Planar Elastic Surface

The goal of this task is to slide a prismatic block over a planar surface along the x_t -axis, while pushing with a given force against the elastic planar surface. The object has three degrees of motion freedom, hence the dimension of the velocity-controlled subspace is $6 - m = 3$ while the dimension of the force-controlled subspace is $m = 3$. The task frame can be chosen attached to the environment, as shown in Fig. 7.5, and the task can be achieved by assigning the desired velocities:

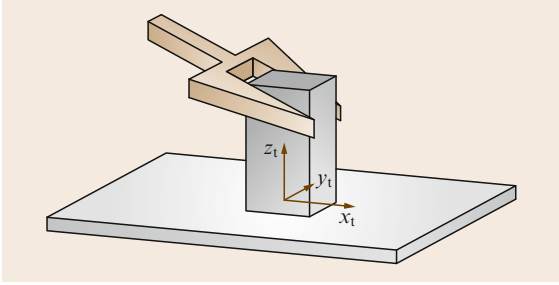


Fig. 7.5 Sliding of a prismatic object on a planar elastic surface

- a nonzero velocity along the x_t -axis
- a zero velocity along the y_t -axis
- a zero angular velocity about the z_t -axis

and the desired forces and torques

- a nonzero force along the z_t -axis
- zero torques about the x_t and y_t axes

Hence, the matrices S_f and S_v can be chosen as

$$S_f = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \quad S_v = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The elements of the 6×6 stiffness matrix K' , corresponding to the partially constrained interaction of the end-effector with the environment, are all zero except those of the 3×3 principal minor K'_m formed by rows 3, 4, 5 and columns 3, 4, 5 of K' , which can be computed as

$$K'_m = \begin{pmatrix} c_{3,3} & c_{3,4} & c_{3,5} \\ c_{4,3} & c_{4,4} & c_{4,5} \\ c_{5,3} & c_{5,4} & c_{5,5} \end{pmatrix}^{-1},$$

where $c_{i,j} = c_{j,i}$ are the elements of the compliance matrix C .

General Contact Model

The task frame concept has proven to be very useful for the specification of a variety of practical robotic tasks. However, it only applies to task geometries with limited complexity, for which separate control modes can be assigned independently to three pure translational and three pure rotational directions along the axes of a single frame. For more complex situations, as in

the case of multiple-point contact, a task frame may not exist and more complex contact models have to be adopted. A possible solution is represented by the *virtual contact manipulator* model, where each individual contact is modeled by a virtual kinematic chain between the manipulated object and the environment, giving the manipulated object (instantaneously) the same motion freedom as the contact. The velocities and force kinematics of the parallel manipulator, formed by the virtual manipulators of all individual contacts, can be derived using the standard kinematics procedures of real manipulators and allow the construction of bases for the twist and wrench spaces of the total motion constraint.

A more general approach, known as *constraint-based task specification* opens up new applications involving complex geometries and/or the use of multiple sensors (force/torque, distance, visual sensors) for controlling different directions in space simultaneously. The concept of task frame is extended to that of multiple *feature frames*. Each of the feature frames makes it possible to model part of the task geometry using translational and rotational directions along the axes of a frame; also, part of the constraints is specified in each of the feature frames. The total model and the total set of constraints are achieved by collecting the partial task and constraints descriptions, each expressed in the individual feature frames.

7.3.4 Sensor-Based Contact Model Estimation

The task specification relies on the definition of the velocity-controlled subspace and of the force-controlled subspace assuming that an accurate model of the contact is available all the time. On the other hand, in most practical implementations, the selection matrices S_f and S_v are not exactly known, however many interaction control strategies turn out to be rather robust against modeling errors. In fact, to cope reliably with these situations is exactly why force control is used. The robustness of the force controller increases if the matrices S_f and S_v can be continuously updated, using motion and/or force measurements, during task execution.

In detail, a nominal model is assumed to be available; when the contact situation evolves differently from what the model predicts, the measured and predicted motion and force will begin to deviate. These small incompatibilities can be measured and can then be used to adapt the model online, using algorithms derived from classical state-space *prediction-correction* estimators such as the Kalman filter.

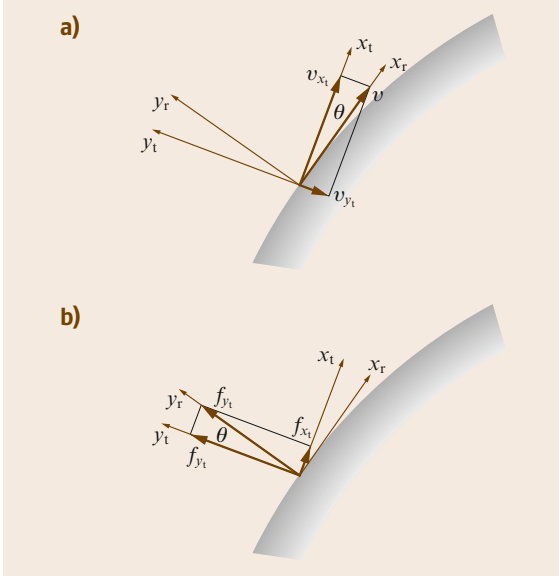


Fig. 7.6a,b Estimation of orientation error: (a) velocity-based approach, (b) force-based approach

Figure 7.6 reports an example of error between nominal and measured motion and force variables, typical of

a two-dimensional contour-following task. The orientation of the contact normal changes if the environment is not planar. Hence an angular error θ appears between the nominal contact normal, aligned to the y_t -axis of the task frame (the frame with axes x_t and y_t), and the real contact normal, aligned to the y_r -axis of the real task frame (the frame with axes x_r and y_r). This angle can be estimated with either velocity or force measurements only.

- *Velocity-based approach*: the executed linear velocity \mathbf{v} , which is tangent to the real contour (aligned to the x_r -axis), does not completely lie along the x_t -axis, but has a small component v_{y_t} along the y_t -axis. The orientation error θ can then be approximated by $\theta = \tan^{-1}(v_{y_t}/v_{x_t})$.
- *Force-based approach*: the measured (ideal) contact force \mathbf{f} does not completely lie along the nominal normal direction, aligned to the y_t -axis, but has a small component f_{x_t} along the x_t -axis. The orientation error θ can then be approximated by $\theta = \tan^{-1}(f_{x_t}/f_{y_t})$.

The velocity-based approach is disturbed by mechanical compliance in the system; the force-based approach is disturbed by contact friction.

7.4 Hybrid Force/Motion Control

The aim of hybrid force/motion control is to split up simultaneous control of both end-effector motion and contact forces into two separate decoupled subproblems. In the following, the main control approaches in the hybrid framework are presented for the cases of both rigid and compliant environments.

7.4.1 Acceleration-Resolved Approach

As for the case of motion control, the acceleration-resolved approach is aimed at decoupling and linearizing the nonlinear robot dynamics at the acceleration level, via an inverse dynamics control law. In the presence of interaction with the environment, a complete decoupling between the force- and velocity-controlled subspaces is sought. The basic idea is that of designing a model-based *inner control loop* to compensate for the nonlinear dynamics of the robot manipulator and decouple the force and velocity subspaces; hence an *outer control loop* is designed to ensure disturbance rejection and tracking of the desired end-effector force and motion.

Rigid Environment

In the case of a rigid environment, the external wrench can be written in the form $\mathbf{h}_e = \mathbf{S}_f \boldsymbol{\lambda}$. The force multiplier $\boldsymbol{\lambda}$ can be eliminated from (7.2) by solving (7.2) for $\dot{\mathbf{v}}_e$ and substituting it into the time derivative of the last equality (7.33). This yields:

$$\boldsymbol{\lambda} = \mathbf{A}_f(\mathbf{q}) \left\{ \mathbf{S}_f^\top \mathbf{A}^{-1}(\mathbf{q}) [\mathbf{h}_c - \boldsymbol{\mu}(\mathbf{q}, \dot{\mathbf{q}})] + \dot{\mathbf{S}}_f^\top \mathbf{v}_e \right\}, \quad (7.49)$$

where $\mathbf{A}_f(\mathbf{q}) = (\mathbf{S}_f^\top \mathbf{A}^{-1} \mathbf{S}_f)^{-1}$ and $\boldsymbol{\mu}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\Gamma} \dot{\mathbf{q}} + \boldsymbol{\eta}$. Therefore the constraint dynamics can be rewritten in the form

$$\mathbf{A}(\mathbf{q}) \dot{\mathbf{v}}_e + \mathbf{S}_f \mathbf{A}_f(\mathbf{q}) \dot{\mathbf{S}}_f^\top \mathbf{v}_e = \mathbf{P}(\mathbf{q}) [\mathbf{h}_c - \boldsymbol{\mu}(\mathbf{q}, \dot{\mathbf{q}})], \quad (7.50)$$

where $\mathbf{P} = \mathbf{I} - \mathbf{S}_f \mathbf{A}_f \mathbf{S}_f^\top \mathbf{A}^{-1}$. Notice that $\mathbf{P} \mathbf{S}_f = \mathbf{0}$, hence the 6×6 matrix \mathbf{P} is a projection matrix that filters out all the end-effector wrenches lying in the range of \mathbf{S}_f . These correspond to wrenches that tend to violate the constraints.

Equation (7.49) reveals that the force multiplier vector λ instantaneously depends also on the applied input wrench \mathbf{h}_c . Hence, by suitably choosing \mathbf{h}_c , it is possible to directly control the m independent components of the end-effector wrench that tend to violate the constraints; these components can be computed from the m force multipliers through (7.29). On the other hand, (7.50) represents a set of six second-order differential equations whose solution, if initialized on the constraints, automatically satisfy (7.27) at all times.

The reduced-order dynamics of the constrained system is described by $6 - m$ second-order equations that are obtained by premultiplying both sides of (7.50) by the matrix \mathbf{S}_v^\top and substituting the acceleration $\dot{\mathbf{v}}_e$ with

$$\dot{\mathbf{v}}_e = \mathbf{S}_v \dot{\mathbf{v}} + \dot{\mathbf{S}}_v \mathbf{v}.$$

The resulting equations are

$$\mathbf{A}_v(q) \dot{\mathbf{v}} = \mathbf{S}_v^\top [\mathbf{h}_c - \boldsymbol{\mu}(q, \dot{q}) - \mathbf{A}(q) \dot{\mathbf{S}}_v \mathbf{v}], \quad (7.51)$$

where $\mathbf{A}_v = \mathbf{S}_v^\top \mathbf{A} \mathbf{S}_v$ and the identities (7.36) and $\mathbf{S}_v^\top \mathbf{P} = \mathbf{S}_v$ have been used. Moreover, expression (7.49) can be rewritten as

$$\lambda = \mathbf{A}_f(q) \mathbf{S}_f^\top \mathbf{A}^{-1}(q) [\mathbf{h}_c - \boldsymbol{\mu}(q, \dot{q}) - \mathbf{A}(q) \dot{\mathbf{S}}_v \mathbf{v}],$$

where the identity $\dot{\mathbf{S}}_f^\top \mathbf{S}_v = -\mathbf{S}_f^\top \dot{\mathbf{S}}_v$ has been exploited.

An inverse-dynamics *inner control loop* can be designed by choosing the control wrench \mathbf{h}_c as

$$\mathbf{h}_c = \mathbf{A}(q) \mathbf{S}_v \boldsymbol{\alpha}_v + \mathbf{S}_f \mathbf{f}_\lambda + \boldsymbol{\mu}(q, \dot{q}) + \mathbf{A}(q) \dot{\mathbf{S}}_v \mathbf{v}, \quad (7.52)$$

where $\boldsymbol{\alpha}_v$ and \mathbf{f}_λ are properly designed control inputs.

Substituting (7.52) into (7.51) and (7.49) yields

$$\begin{aligned} \dot{\mathbf{v}} &= \boldsymbol{\alpha}_v \\ \lambda &= \mathbf{f}_\lambda, \end{aligned}$$

showing that control law (7.52) allows complete decoupling between the force- and velocity-controlled subspaces.

It is worth noticing that, for the implementation of the control law (7.52), the constraint (7.27) as well as (7.38) defining the vector of the configuration variables for the constrained system are not required, provided that the matrices \mathbf{S}_f and \mathbf{S}_v are known or estimated online. In these cases, the task can easily be assigned by specifying a desired force, in terms of the vector $\lambda_d(t)$, and a desired velocity, in terms of the vector $\mathbf{v}_d(t)$; moreover, a force/velocity control is implemented.

The desired force $\lambda_d(t)$ can be achieved by setting

$$\mathbf{f}_\lambda = \lambda_d(t), \quad (7.53)$$

but this choice is very sensitive to disturbance forces, since it contains no force feedback. Alternative choices are

$$\mathbf{f}_\lambda = \lambda_d(t) + \mathbf{K}_{P\lambda} [\lambda_d(t) - \lambda(t)], \quad (7.54)$$

or

$$\mathbf{f}_\lambda = \lambda_d(t) + \mathbf{K}_{I\lambda} \int_0^t [\lambda_d(\tau) - \lambda(\tau)] d\tau, \quad (7.55)$$

where $\mathbf{K}_{P\lambda}$ and $\mathbf{K}_{I\lambda}$ are suitable positive-definite matrix gains. The proportional feedback is able to reduce the force error due to disturbance forces, while the integral action is able to compensate for constant bias disturbances.

The implementation of force feedback requires the computation of the force multiplier λ from the measurement of the end-effector wrench \mathbf{h}_e , which can be achieved by using (7.31).

Velocity control is achieved by setting

$$\begin{aligned} \boldsymbol{\alpha}_v &= \dot{\mathbf{v}}_d(t) + \mathbf{K}_{Pv} [\mathbf{v}_d(t) - \mathbf{v}(t)] \\ &\quad + \mathbf{K}_{Iv} \int_0^t [\mathbf{v}_d(\tau) - \mathbf{v}(\tau)] d\tau, \end{aligned} \quad (7.56)$$

where \mathbf{K}_{Pv} and \mathbf{K}_{Iv} are suitable matrix gains. It is straightforward to show that asymptotic tracking of $\mathbf{v}_d(t)$ and $\dot{\mathbf{v}}_d(t)$ is ensured with exponential convergence for any choice of positive-definite matrices \mathbf{K}_{Pv} and \mathbf{K}_{Iv} .

The computation of the vector \mathbf{v} from the available measurements can be achieved using (7.37), where the end-effector twist is computed from joint position and velocity measurements through (7.1).

Equations (7.54) or (7.55) and (7.56) represent the *outer control loop* ensuring force/velocity control and disturbance rejection.

When equations (7.27) and (7.38) are known, the matrices \mathbf{S}_f and \mathbf{S}_v can be computed according to (7.30) and (7.40) and a *force/position* control can be designed by specifying a desired force $\lambda_d(t)$ and a desired position $\mathbf{r}_d(t)$.

Force control can be designed as above, while position control can be achieved by setting

$$\boldsymbol{\alpha}_v = \ddot{\mathbf{r}}_d(t) + \mathbf{K}_{Dr} [\dot{\mathbf{r}}_d(t) - \mathbf{v}(t)] + \mathbf{K}_{Pr} [\mathbf{r}_d(t) - \mathbf{r}(t)].$$

Asymptotic tracking of $\mathbf{r}_d(t)$, $\dot{\mathbf{r}}_d(t)$ and $\ddot{\mathbf{r}}_d(t)$ is ensured with exponential convergence for any choice of positive-definite matrices \mathbf{K}_{Dr} and \mathbf{K}_{Pr} . The vector \mathbf{r} required for position feedback can be computed from joint position measurements via (7.38).

Compliant Environment

In the case of a compliant environment, according to the decomposition (7.42) of the end-effector displacement, the end-effector twist can be decomposed as

$$\mathbf{v}_e = \mathbf{S}_v \mathbf{v} + \mathbf{C}' \mathbf{S}_f \dot{\boldsymbol{\lambda}}, \quad (7.57)$$

where the first term is a twist of freedom, the second term is a twist of constraint, the vector \mathbf{v} is defined as in (7.40), and \mathbf{C}' is defined in (7.48). Assuming a constant contact geometry and compliance, i.e., $\dot{\mathbf{S}}_v = \mathbf{0}$, $\dot{\mathbf{C}}' = \mathbf{0}$, and $\dot{\mathbf{S}}_f = \mathbf{0}$, a similar decomposition holds in terms of acceleration

$$\dot{\mathbf{v}}_e = \mathbf{S}_v \dot{\mathbf{v}} + \mathbf{C}' \mathbf{S}_f \ddot{\boldsymbol{\lambda}}. \quad (7.58)$$

An inverse-dynamics control law (7.15) can be adopted, resulting in the closed loop (7.16), where $\boldsymbol{\alpha}$ is a properly designed control input.

In view of the acceleration decomposition (7.58), the choice

$$\boldsymbol{\alpha} = \mathbf{S}_v \boldsymbol{\alpha}_v + \mathbf{C}' \mathbf{S}_f \mathbf{f}_\lambda \quad (7.59)$$

allows decoupling of the force control from velocity control. In fact, substituting (7.58) and (7.59) into (7.16) and premultiplying both sides of the resulting equation once by \mathbf{S}_v^\top and once by \mathbf{S}_f^\top , the following decoupled equations can be derived

$$\dot{\mathbf{v}} = \boldsymbol{\alpha}_v \quad (7.60)$$

$$\ddot{\boldsymbol{\lambda}} = \mathbf{f}_\lambda. \quad (7.61)$$

Hence, by choosing $\boldsymbol{\alpha}_v$ according to (7.56) as for the rigid environment case, asymptotic tracking of a desired velocity $\mathbf{v}_d(t)$ and acceleration $\dot{\mathbf{v}}_d(t)$ is ensured, with exponential convergence. The control input \mathbf{f}_λ can be chosen as

$$\begin{aligned} \mathbf{f}_\lambda &= \ddot{\boldsymbol{\lambda}}_d(t) + \mathbf{K}_{D\lambda} [\dot{\boldsymbol{\lambda}}_d(t) - \dot{\boldsymbol{\lambda}}(t)] \\ &\quad + \mathbf{K}_{P\lambda} [\boldsymbol{\lambda}_d(t) - \boldsymbol{\lambda}(t)], \end{aligned} \quad (7.62)$$

ensuring asymptotic tracking of a desired force trajectory $(\boldsymbol{\lambda}_d(t), \dot{\boldsymbol{\lambda}}_d(t), \ddot{\boldsymbol{\lambda}}_d(t))$ with exponential convergence for any choice of positive-definite matrices $\mathbf{K}_{D\lambda}$ and $\mathbf{K}_{P\lambda}$.

Differently from the rigid environment case, feedback of $\dot{\boldsymbol{\lambda}}$ is required for the implementation of the force control law (7.62). This quantity could be computed from end-effector wrench measurements \mathbf{h}_e as

$$\dot{\boldsymbol{\lambda}} = \mathbf{S}_f^\top \dot{\mathbf{h}}_e.$$

However, since the wrench measurement signal is often noisy, the feedback of $\dot{\boldsymbol{\lambda}}$ is often replaced by

$$\dot{\boldsymbol{\lambda}} = \mathbf{S}_f^\top \mathbf{K}' \mathbf{J}(\mathbf{q}) \dot{\mathbf{q}}, \quad (7.63)$$

where joint velocities are measured using tachometers or computed from joint positions via numerical differentiation and \mathbf{K}' is the positive-semidefinite stiffness matrix (7.47) describing the partially constrained interaction. For the computation of (7.63), only knowledge (or an estimate) of \mathbf{K}' is required, and not that of the full stiffness matrix \mathbf{K} . Also, the implementation of the control law (7.59) requires knowledge (or an estimate) of the compliance matrix \mathbf{C}' of the partially constrained interaction and not that of the full compliance matrix \mathbf{C} .

If the contact geometry is known, but only an estimate of the stiffness/compliance of the environment is available, the control law (7.59), with (7.62), may still guarantee the convergence of the force error, if a constant desired force $\boldsymbol{\lambda}_d$ is assigned. In this case, the control law (7.59) has the form

$$\boldsymbol{\alpha} = \mathbf{S}_v \boldsymbol{\alpha}_v + \hat{\mathbf{C}}' \mathbf{S}_f \mathbf{f}_\lambda,$$

where $\hat{\mathbf{C}}' = (\mathbf{I} - \mathbf{P}_v) \hat{\mathbf{C}}$ and $\hat{\mathbf{C}}$ is an estimate of the compliance matrix. Hence, (7.60) still holds, while, in lieu of (7.61), the following equality can be found

$$\ddot{\boldsymbol{\lambda}} = \mathbf{L}_f \mathbf{f}_\lambda$$

where $\mathbf{L}_f = (\mathbf{S}_f^\top \mathbf{C} \mathbf{S}_f)^{-1} \mathbf{S}_f^\top \hat{\mathbf{C}} \mathbf{S}_f$ is a nonsingular matrix. Thus, the force- and velocity-controlled subspaces remain decoupled and the velocity control law (7.56) does not need to be modified. On the other hand, if the feedback of the time derivative of $\boldsymbol{\lambda}$ is computed using (7.63), only an estimate $\hat{\boldsymbol{\lambda}}$ can be obtained. Using (7.63), (7.57) and (7.46), the following equality can be found

$$\dot{\hat{\boldsymbol{\lambda}}} = \mathbf{L}_f^{-1} \dot{\boldsymbol{\lambda}}.$$

Therefore, computing the force control law \mathbf{f}_λ as in (7.62) with a constant $\boldsymbol{\lambda}_d$, $\hat{\boldsymbol{\lambda}}$ in lieu of $\boldsymbol{\lambda}$ and $\mathbf{K}_{D\lambda} = \mathbf{K}_{D\lambda} \mathbf{I}$, the dynamics of the closed-loop system is

$$\ddot{\hat{\boldsymbol{\lambda}}} + \mathbf{K}_{D\lambda} \dot{\hat{\boldsymbol{\lambda}}} + \mathbf{L}_f \mathbf{K}_{P\lambda} \hat{\boldsymbol{\lambda}} = \mathbf{L}_f \mathbf{K}_{P\lambda} \boldsymbol{\lambda}_d,$$

showing that exponential asymptotic stability of the equilibrium $\boldsymbol{\lambda} = \boldsymbol{\lambda}_d$ can be ensured, also in the presence of the uncertain matrix \mathbf{L}_f , with a suitable choice of the gains $\mathbf{K}_{D\lambda}$ and $\mathbf{K}_{P\lambda}$.

7.4.2 Passivity-Based Approach

The passivity-based approach exploits the passivity properties of the dynamic model of the manipulator, which hold also for the constrained dynamic model (7.2). It can be easily shown that the choice of the matrix

$C(q, \dot{q})$ that guarantees the skew symmetry of the matrix $\dot{H}(q) - 2C(q, \dot{q})$ in the joint space, also makes the matrix $\dot{A}(q) - 2\Gamma(q, \dot{q})$ skew symmetric. This fundamental property of Lagrangian systems is at the base of passivity-based control algorithms.

Rigid Environment

The control wrench h_c can be chosen as

$$h_c = A(q)S_v \dot{v}_r + \Gamma'(q, \dot{q})v_r + (S_v^\dagger)^\top K_v(v_r - v) + \eta(q) + S_f f_\lambda, \quad (7.64)$$

where $\Gamma'(q, \dot{q}) = \Gamma S_v + A\dot{S}_v$, K_v is a suitable symmetric and positive-definite matrix, and v_r and f_λ are properly designed control inputs.

Substituting (7.64) into (7.2) yields

$$A(q)S_v \dot{s}_v + \Gamma'(q, \dot{q})s_v + (S_v^\dagger)^\top K_v s_v + S_f(f_\lambda - \lambda) = 0 \quad (7.65)$$

with $\dot{s}_v = \dot{v}_r - \dot{v}$ and $s_v = v_r - v$, showing that the closed-loop system remains nonlinear and coupled.

Premultiplying both sides of (7.65) by the matrix S_v , the following expression for the reduced-order dynamics is achieved

$$A_v(q)\dot{s}_v + \Gamma_v(q, \dot{q})s_v + K_v s_v = 0, \quad (7.66)$$

with $\Gamma_v = S_v^\top \Gamma(q, \dot{q})S_v + S_v^\top A(q)\dot{S}_v$; it can easily be shown that the skew symmetry of the matrix $\dot{A}(q) - 2\Gamma(q, \dot{q})$ implies that the matrix $\dot{A}_v(q) - 2\Gamma_v(q, \dot{q})$ is skew symmetric as well.

On the other hand, premultiplying both sides of (7.65) by the matrix $S_f^\top A^{-1}(q)$, the following expression for the force dynamics can be found

$$f_\lambda - \lambda = -A_f(q)S_f^\top \cdot A^{-1}(q)[\Gamma'(q, \dot{q}) - (S_v^\dagger)^\top K_v]s_v, \quad (7.67)$$

showing that the force multiplier λ instantaneously depends on the control input f_λ but also on the error s_v in the velocity-controlled subspace.

The asymptotic stability of the reduced-order system (7.66) can be ensured with the choices

$$\dot{v}_r = \dot{v}_d + \alpha \Delta v, \quad (7.68)$$

$$v_r = v_d + \alpha \Delta x_v, \quad (7.69)$$

where α is a positive gain, \dot{v}_d and v_d are the desired acceleration and velocity, respectively, $\Delta v = v_d - v$, and

$$\Delta x_v = \int_0^t \Delta v(\tau) d\tau.$$

The stability proof is based on the positive-definite Lyapunov function

$$V = \frac{1}{2} s_v^\top A_v(q) s_v + \alpha \Delta x_v^\top K_v \Delta x_v,$$

whose time derivative along the trajectories of (7.66),

$$\dot{V} = -\Delta v^\top K_v \Delta v - \alpha^2 \Delta x_v^\top K_v \Delta x_v,$$

is a definite semi-negative function. Hence, $\Delta v = 0$, $\Delta x_v = 0$, and $s_v = 0$ asymptotically. Therefore, tracking of the desired velocity $v_d(t)$ is ensured. Moreover, the right-hand side of (7.67) remains bounded and vanishes asymptotically. Hence, tracking of the desired force $\lambda_d(t)$ can be ensured by setting f_λ as for the acceleration-resolved approach, according to the choices (7.53), (7.54), or (7.55).

Notice that position control can be achieved if a desired position $r_d(t)$ is assigned for the vector r in (7.38), provided that the matrices S_f and S_v are computed according to (7.30) and (7.40), and the vectors $\dot{v}_d = \ddot{r}_d$, $v_d = \dot{r}_d$, and $\Delta x_v = r_d - r$ are used in (7.68) and (7.69).

Compliant Environment

The control wrench h_c can be chosen as

$$h_c = A(q)\dot{v}_r + \Gamma(q, \dot{q})v_r + K_s(v_r - v_e) + h_e + \eta(q), \quad (7.70)$$

where K_s is a suitable symmetric positive-definite matrix while v_r and its time derivative \dot{v}_r are chosen as

$$v_r = v_d + \alpha \Delta x,$$

$$\dot{v}_r = \dot{v}_d + \alpha \Delta v,$$

where α is the positive gain, v_d and its time derivative \dot{v}_d are properly designed control inputs, $\Delta v = v_d - v_e$, and

$$\Delta x = \int_0^t \Delta v d\tau.$$

Substituting (7.70) into (7.2) yields

$$A(q)\dot{s} + \Gamma(q, \dot{q})s + K_s s = 0 \quad (7.71)$$

with $\dot{s} = \dot{v}_r - \dot{v}_e$ and $s = v_r - v_e$.

The asymptotic stability of system (7.71) can be ensured by setting:

$$v_d = S_v v_d + C' S_f \dot{\lambda}_d,$$

where $v_d(t)$ is a desired velocity trajectory and $\lambda_d(t)$ is the desired force trajectory. The stability proof is based on the positive-definite Lyapunov function

$$V = \frac{1}{2} s^\top A(q) s + \alpha \Delta x^\top K_s \Delta x,$$

whose time derivative along the trajectories of (7.71),

$$\dot{V} = -\Delta \mathbf{v}^\top \mathbf{K}_s \Delta \mathbf{v} - \alpha^2 \Delta \mathbf{x}^\top \mathbf{K}_s \Delta \mathbf{x},$$

is a negative-definite function. Hence, $\Delta \mathbf{v} = \mathbf{0}$ and $\Delta \mathbf{x} = \mathbf{0}$, asymptotically. In the case of constant contact geometry and stiffness, the following equalities hold

$$\begin{aligned} \Delta \mathbf{v} &= \mathbf{S}_v(\mathbf{v}_d - \mathbf{v}) + \mathbf{C}' \mathbf{S}_f(\dot{\lambda}_d - \dot{\lambda}), \\ \Delta \mathbf{x} &= \mathbf{S}_v \int_0^t (\mathbf{v}_d - \mathbf{v}) d\tau + \mathbf{C}' \mathbf{S}_f(\lambda_d - \lambda), \end{aligned}$$

showing that both the velocity and force tracking errors, belonging to reciprocal subspaces, converge asymptotically to zero.

7.4.3 Velocity-Resolved Approach

The acceleration-resolved approach, as well as the passivity-based approach, require modification of the current industrial robot controllers. As for the case of impedance control, if the contact is sufficiently compliant, the closed-loop dynamics of a motion-controlled robot can be approximated by (7.26), corresponding to a velocity-resolved control.

To achieve force and velocity control, according to the end-effector twist decomposition (7.57), the control input \mathbf{v}_r can be chosen as

$$\mathbf{v}_r = \mathbf{S}_v \mathbf{v}_v + \mathbf{C}' \mathbf{S}_f \mathbf{f}_\lambda, \quad (7.72)$$

with

$$\mathbf{v}_v = \mathbf{v}_d(t) + \mathbf{K}_{Iv} \int_0^t [\mathbf{v}_d(\tau) - \mathbf{v}(\tau)] d\tau, \quad (7.73)$$

and

$$\mathbf{f}_\lambda = \dot{\lambda}_d(t) + \mathbf{K}_{P\lambda} [\lambda_d(t) - \lambda(t)], \quad (7.74)$$

where \mathbf{K}_{Iv} and $\mathbf{K}_{P\lambda}$ are suitable symmetric and positive-definite matrix gains. Decoupling between velocity- and force-controlled subspaces and exponential asymptotic stability of the closed-loop system can be proven as for the acceleration-resolved approach. Also, since the force error has second-order dynamics, an integral action can be added to (7.74) to improve the disturbance rejection capabilities, i. e.:

$$\begin{aligned} \mathbf{f}_\lambda &= \dot{\lambda}_d(t) + \mathbf{K}_{P\lambda} [\lambda_d(t) - \lambda(t)] \\ &\quad + \mathbf{K}_{I\lambda} \int_0^t [\lambda_d(\tau) - \lambda(\tau)] d\tau, \end{aligned} \quad (7.75)$$

and the exponential asymptotic stability is guaranteed if the matrices $\mathbf{K}_{P\lambda}$ and $\mathbf{K}_{I\lambda}$ are symmetric and positive definite.

If an estimate $\hat{\mathbf{C}}$ of the stiffness matrix of the environment is used in (7.72), as for the acceleration-resolved approach, the exponential convergence of λ to a constant λ_d can still be ensured for both (7.74) and (7.75).

In some applications, besides the stiffness matrix, also the geometry of the environment is uncertain. In these cases, a force/motion control law similar to (7.72) can be implemented, without using the selection matrices \mathbf{S}_v and \mathbf{S}_f to separate the force-controlled subspace from the velocity-controlled subspace. The motion control law can be set as in (7.73), but using full velocity feedback. Also, the force control law can be set as in (7.75), but using full force and moment feedback. That is, both motion control and force control are applied in all directions of the six-dimensional (6-D) space. The resulting control, known as *force control with feedforward motion* or *parallel force/position control* guarantees force regulation at the expense of position errors along the constrained task directions, thanks to the dominance of the force controller over the position controller ensured by the presence of the integral action on the force error.

7.5 Conclusions and Further Reading

This chapter has summarized the main approaches to force control in a unifying perspective. However, there are many aspects that have not been considered and that must be carefully taken into account when dealing with interaction robotic tasks. The two major paradigms of force control (impedance and hybrid force/motion control) are based on several simplifying assumptions that are only partially satisfied in practical implemen-

tations. In fact, the performance of a force-controlled robotic system depends on the interaction with a changing environment which is very difficult to model and identify correctly. A general contact situation is far from being completely predictable, not only quantitatively, but also qualitatively: the contact configuration can change abruptly, or be of a different type than expected. Hence, the standard performance indices used

to evaluate a control system, i.e., stability, bandwidth, accuracy, and robustness, cannot be defined by considering the robotic system alone, as for the case of robot motion control, but must be always referred to the particular contact situation at hand. Also, a classification of all these different situations is not easy, especially in the case of dynamics environments and when the task involves multiple contacts acting in parallel.

Due to the inherent complexity of the force control problem, a large number of research papers on this topic have been published in the past three decades. A description of the state of the art of the first decade is provided in [7.20], whereas the progress of the second decade is surveyed in [7.21] and [7.22]. More recently, two monographs on force control [7.23, 24] have appeared. In the following, a list of references is provided, where more details on the arguments presented in the chapter, as well as topics not covered here, can be found.

7.5.1 Indirect Force Control

The concept of generalized spring and damping for force control in joint coordinates was originally proposed in [7.3] and the implementation discussed in [7.10]. Stiffness control in Cartesian coordinates was proposed in [7.9]. Devices based on the remote center of compliance were discussed in [7.25] for successful mating of rigid parts. The original idea of a mechanical impedance model used for controlling the interaction between the manipulator and the environment is presented in [7.7], and a similar formulation is given in [7.8]. Stability of impedance control was analyzed in [7.26] and the problems of interaction with a stiff environment were considered in [7.27].

Adaptive impedance control algorithms [7.28, 29] have been proposed to overcome uncertainties in the dynamic parameters of the robot manipulator, while robust control schemes can be found in [7.30]. Impedance control has also been used in the hybrid force/motion control framework [7.31].

A reference work on modeling 6-DOF (spatial) stiffness is [7.32], while the properties of spatial compliance have been analyzed in detail in [7.33–35]; a 6-DOF variable compliant wrist was proposed in [7.36], while several studies concerning the construction of programmed compliances, optimized for specific tasks, have been proposed [7.37, 38]. The energy-based approach to derive a spatial impedance was introduced in [7.39], using rotation matrices; various 6-DOF impedance control schemes based on different representations of end-effector orientation, including the unit

quaternion, can be found in [7.40]. The quaternion-based formulation is extended to the case of non-block-diagonal stiffness matrix in [7.41]. A rigorous treatment of spatial impedance control in a passivity framework can be found in [7.42].

7.5.2 Task Specification

The concepts of natural and artificial constraints and of compliance frame were introduced in [7.11]. These ideas have been systematically developed in [7.12, 43] within the *task frame formalism*. Theoretical issues on the reciprocity of generalized force and velocity directions are discussed in [7.44, 45], while invariance in computing the generalized inverse in robotics is addressed in [7.46]. The issue of partially constrained tasks is considered in [7.47], where the models of positive-semidefinite stiffness and compliance matrices are developed. The problem of estimating geometric uncertainties is considered in [7.48, 49], as well as the issue of linking constraint-based task specification with real-time task execution control. This approach is generalized in [7.50], where a systematic constraint-based methodology to specify complex tasks is presented.

7.5.3 Hybrid Force/Motion Control

Early works on force control can be found in [7.10]. The original hybrid force/position control concept was introduced in [7.13], based on the natural and artificial constraint task formulation [7.11]. The explicit inclusion of the manipulator dynamic model was presented in [7.17], and a systematic approach to modeling the interaction with a dynamic environment was developed in [7.51]. The constrained formulation with inverse dynamic controllers is treated in [7.14, 52] in the Cartesian space as well as in [7.15] joint space. The constrained approach was also used in [7.16] with a controller based on linearized equations. The invariance problems pointed out in [7.45] were correctly addressed, among other papers, in [7.44, 53]. Transposition of model-based schemes from unconstrained motion control to constrained cases was accomplished for adaptive control in [7.18, 54, 55] and for robust control in [7.56].

Approaches designed to cope with uncertainties in the environment geometry are the force control with feedforward motion scheme proposed in [7.2] and the parallel force/position control [7.19], based on the concept of dominance of force control on motion control, thanks to the use of an integral action on the force error. A parallel force/position regulator was de-

veloped in [7.57]. The integral action for removing steady-state force errors has traditionally been used; its stability was proven in [7.58], while robustness with respect to force measurement delays was investigated in [7.59, 60].

It has generally been recognized that force control may cause unstable behavior during contact with environment. Dynamic models for explaining this phenomenon were introduced in [7.61] and experimental

investigations can be found in [7.62] and [7.63]. Moreover, control schemes are usually derived on the assumption that the manipulator end-effector is in contact with the environment and that this contact is not lost. Impact phenomena may occur and deserve careful consideration, and there is a need for global analysis of control schemes including the transition from noncontact to contact situations and vice versa, see e.g., [7.64–66].

References

- 7.1 T.L. De Fazio, D.S. Seltzer, D.E. Whitney: The instrumented remote center of compliance, *Ind. Robot* **11**(4), 238–242 (1984)
- 7.2 J. De Schutter, H. Van Brussel: Compliant robot motion II. A control approach based on external control loops, *Int. J. Robot. Res.* **7**(4), 18–33 (1988)
- 7.3 I. Nevins, D.E. Whitney: The force vector assembler concept, First CISM-IFTOMM Symp. Theory Pract. Robot. Manip. (Udine 1973)
- 7.4 M.T. Mason, J.K. Salisbury: *Robot Hands and Mechanics of Manipulation* (MIT Press, Cambridge 1985)
- 7.5 J.Y.S. Luh, W.D. Fisher, R.P.C. Paul: Joint torque control by direct feedback for industrial robots, *IEEE Trans. Autom. Contr.* **28**, 153–161 (1983)
- 7.6 G. Hirzinger, N. Sporer, A. Albu-Shäffer, M. Hähnele, R. Krenn, A. Pascucci, R. Schedl: DLR's torque-controlled light weight robot III – are we reaching the technological limits now?, *IEEE Int. Conf. Robot. Autom.* (Washington 2002) pp. 1710–1716
- 7.7 N. Hogan: Impedance control: an approach to manipulation: parts I–III, *ASME J. Dyn. Syst. Meas. Contr.* **107**, 1–24 (1985)
- 7.8 H. Kazerooni, T.B. Sheridan, P.K. Houpt: Robust compliant motion for manipulators. Part I: the fundamental concepts of compliant motion, *IEEE J. Robot. Autom.* **2**, 83–92 (1986)
- 7.9 J.K. Salisbury: Active stiffness control of a manipulator in Cartesian coordinates, 19th IEEE Conf. Decis. Contr. (Albuquerque, 1980) pp. 95–100
- 7.10 D.E. Whitney: Force feedback control of manipulator fine motions, *ASME J. Dyn. Syst. Meas. Contr.* **99**, 91–97 (1977)
- 7.11 M.T. Mason: Compliance and force control for computer controlled manipulators, *IEEE Trans. Syst. Man Cybern.* **11**, 418–432 (1981)
- 7.12 J. De Schutter, H. Van Brussel: Compliant robot motion I. A formalism for specifying compliant motion tasks, *Int. J. Robot. Res.* **7**(4), 3–17 (1988)
- 7.13 M.H. Raibert, J.J. Craig: Hybrid position/force control of manipulators, *ASME J. Dyn. Syst. Meas. Contr.* **103**, 126–133 (1981)
- 7.14 T. Yoshikawa: Dynamic hybrid position/force control of robot manipulators – description of hand constraints and calculation of joint driving force, *IEEE J. Robot. Autom.* **3**, 386–392 (1987)
- 7.15 N.H. McClamroch, D. Wang: Feedback stabilization and tracking of constrained robots, *IEEE Trans. Autom. Contr.* **33**, 419–426 (1988)
- 7.16 J.K. Mills, A.A. Goldenberg: Force and position control of manipulators during constrained motion tasks, *IEEE Trans. Robot. Autom.* **5**, 30–46 (1989)
- 7.17 O. Khatib: A unified approach for motion and force control of robot manipulators: the operational space formulation, *IEEE J. Robot. Autom.* **3**, 43–53 (1987)
- 7.18 L. Villani, C. Canudas de Wit, B. Brogliato: An exponentially stable adaptive control for force and position tracking of robot manipulators, *IEEE Trans. Autom. Contr.* **44**, 798–802 (1999)
- 7.19 S. Chiaverini, L. Sciavicco: The parallel approach to force/position control of robotic manipulators, *IEEE Trans. Robot. Autom.* **9**, 361–373 (1993)
- 7.20 D.E. Whitney: Historical perspective and state of the art in robot force control, *Int. J. Robot. Res.* **6**(1), 3–14 (1987)
- 7.21 M. Vukobratović, Y. Nakamura: Force and contact control in robotic systems., *Tutorial IEEE Int. Conf. Robot. Autom.* (Atlanta 1993)
- 7.22 J. De Schutter, H. Bruyninckx, W.H. Zhu, M.W. Spong: Force control: a bird's eye view. In: *Control Problems in Robotics and Automation*, ed. by K.P. Valavanis, B. Siciliano (Springer, Berlin, Heidelberg 1998) pp. 1–17
- 7.23 D.M. Gorinevski, A.M. Formal'sky, A.Yu. Schneider: *Force Control of Robotics Systems* (CRC Press, Boca Raton 1997)
- 7.24 B. Siciliano, L. Villani: *Robot Force Control* (Kluwer Academic Publishers, Boston 1999)
- 7.25 D.E. Whitney: Quasi-static assembly of compliantly supported rigid parts, *ASME J. Dyn. Syst. Meas. Contr.* **104**, 65–77 (1982)
- 7.26 N. Hogan: On the stability of manipulators performing contact tasks, *IEEE J. Robot. Autom.* **4**, 677–686 (1988)
- 7.27 H. Kazerooni: Contact instability of the direct drive robot when constrained by a rigid environment, *IEEE Trans. Autom. Contr.* **35**, 710–714 (1990)

- 7.28 R. Kelly, R. Carelli, M. Amestegui, R. Ortega: Adaptive impedance control of robot manipulators, *IASTED Int. J. Robot. Autom.* **4**(3), 134–141 (1989)
- 7.29 R. Colbaugh, H. Seraji, K. Glass: Direct adaptive impedance control of robot manipulators, *J. Robot. Syst.* **10**, 217–248 (1993)
- 7.30 Z. Lu, A.A. Goldenberg: Robust impedance control and force regulation: theory and experiments, *Int. J. Robot. Res.* **14**, 225–254 (1995)
- 7.31 R.J. Anderson, M.W. Spong: Hybrid impedance control of robotic manipulators, *IEEE J. Robot. Autom.* **4**, 549–556 (1988)
- 7.32 J. Lončarić: Normal forms of stiffness and compliance matrices, *IEEE J. Robot. Autom.* **3**, 567–572 (1987)
- 7.33 T. Patterson, H. Lipkin: Structure of robot compliance, *ASME J. Mech. Design* **115**, 576–580 (1993)
- 7.34 E.D. Fasse, P.C. Breedveld: Modelling of elastically coupled bodies: part I – General theory and geometric potential function method, *ASME J. Dyn. Syst. Meas. Contr.* **120**, 496–500 (1998)
- 7.35 E.D. Fasse, P.C. Breedveld: Modelling of elastically coupled bodies: part II – Exponential and generalized coordinate method, *ASME J. Dyn. Syst. Meas. Contr.* **120**, 501–506 (1998)
- 7.36 R.L. Hollis, S.E. Salcudean, A.P. Allan: A six-degree-of-freedom magnetically levitated variable compliance fine-motion wrist: design, modeling and control, *IEEE Trans. Robot. Autom.* **7**, 320–333 (1991)
- 7.37 M.A. Peshkin: Programmed compliance for error corrective assembly, *IEEE Trans. Robot. Autom.* **6**, 473–482 (1990)
- 7.38 J.M. Shimmels, M.A. Peshkin: Admittance matrix design for force-guided assembly, *IEEE Trans. Robot. Autom.* **8**, 213–227 (1992)
- 7.39 E.D. Fasse, J.F. Broenink: A spatial impedance controller for robotic manipulation, *IEEE Trans. Robot. Autom.* **13**, 546–556 (1997)
- 7.40 F. Caccavale, C. Natale, B. Siciliano, L. Villani: Six-DOF impedance control based on angle/axis representations, *IEEE Trans. Robot. Autom.* **15**, 289–300 (1999)
- 7.41 F. Caccavale, C. Natale, B. Siciliano, L. Villani: Robot impedance control with nondiagonal stiffness, *IEEE Trans. Autom. Contr.* **44**, 1943–1946 (1999)
- 7.42 S. Stramigioli: *Modeling and IPC Control of Interactive Mechanical Systems – A Coordinate Free Approach*, Lecture Notes in Control and Information Sciences (Springer, London 2001)
- 7.43 H. Bruyninckx, J. De Schutter: Specification of Force-controlled actions in the “task frame formalism” – a synthesis, *IEEE Trans. Robot. Autom.* **12**, 581–589 (1996)
- 7.44 H. Lipkin, J. Duffy: Hybrid twist and wrench control for a robotic manipulator, *ASME J. Mech. Trans. Autom. Des.* **110**, 138–144 (1988)
- 7.45 J. Duffy: The fallacy of modern hybrid control theory that is based on ‘orthogonal complements’ of twist and wrench spaces, *J. Robot. Syst.* **7**, 139–144 (1990)
- 7.46 K.L. Doty, C. Melchiorri, C. Bonivento: A theory of generalized inverses applied to robotics, *Int. J. Robot. Res.* **12**, 1–19 (1993)
- 7.47 T. Patterson, H. Lipkin: Duality of constrained elastic manipulation, *IEEE Conf. Robot. Autom.* (Sacramento 1991) pp. 2820–2825
- 7.48 J. De Schutter, H. Bruyninckx, S. Dutré, J. De Geeter, J. Katupitiya, S. Demey, T. Lefebvre: Estimation first-order geometric parameters and monitoring contact transitions during force-controlled compliant motions, *Int. J. Robot. Res.* **18**(12), 1161–1184 (1999)
- 7.49 T. Lefebvre, H. Bruyninckx, J. De Schutter: Polyedral contact formation identification for autonomous compliant motion, *IEEE Trans. Robot. Autom.* **19**, 26–41 (2007)
- 7.50 J. De Schutter, T. De Laet, J. Rutgeerts, W. Decré, R. Smits, E. Aerbeliën, K. Claes, H. Bruyninckx: Constraint-based task specification and estimation for sensor-based robot systems in the presence of geometric uncertainty, *Int. J. Robot. Res.* **26**(5), 433–455 (2007)
- 7.51 A. De Luca, C. Manes: Modeling robots in contact with a dynamic environment, *IEEE Trans. Robot. Autom.* **10**, 542–548 (1994)
- 7.52 T. Yoshikawa, T. Sugie, N. Tanaka: Dynamic hybrid position/force control of robot manipulators – controller design and experiment, *IEEE J. Robot. Autom.* **4**, 699–705 (1988)
- 7.53 J. De Schutter, D. Torfs, H. Bruyninckx, S. Dutré: Invariant hybrid force/position control of a velocity controlled robot with compliant end effector using modal decoupling, *Int. J. Robot. Res.* **16**(3), 340–356 (1997)
- 7.54 R. Lozano, B. Brogliato: Adaptive hybrid force-position control for redundant manipulators, *IEEE Trans. Autom. Contr.* **37**, 1501–1505 (1992)
- 7.55 L.L. Whitcomb, S. Arimoto, T. Naniwa, F. Ozaki: Adaptive model-based hybrid control if geometrically constrained robots, *IEEE Trans. Robot. Autom.* **13**, 105–116 (1997)
- 7.56 B. Yao, S.P. Chan, D. Wang: Unified formulation of variable structure control schemes for robot manipulators, *IEEE Trans. Autom. Contr.* **39**, 371–376 (1992)
- 7.57 S. Chiaverini, B. Siciliano, L. Villani: Force/position regulation of compliant robot manipulators, *IEEE Trans. Autom. Contr.* **39**, 647–652 (1994)
- 7.58 J.T.-Y. Wen, S. Murphy: Stability analysis of position and force control for robot arms, *IEEE Trans. Autom. Contr.* **36**, 365–371 (1991)
- 7.59 R. Volpe, P. Khosla: A theoretical and experimental investigation of explicit force control strategies for manipulators, *IEEE Trans. Autom. Contr.* **38**, 1634–1650 (1993)

- 7.60 L.S. Wilfinger, J.T. Wen, S.H. Murphy: Integral force control with robustness enhancement, *IEEE Contr. Syst. Mag.* **14**(1), 31–40 (1994)
- 7.61 S.D. Eppinger, W.P. Seering: Introduction to dynamic models for robot force control, *IEEE Contr. Syst. Mag.* **7**(2), 48–52 (1987)
- 7.62 C.H. An, J.M. Hollerbach: The role of dynamic models in Cartesian force control of manipulators, *Int. J. Robot. Res.* **8**(4), 51–72 (1989)
- 7.63 R. Volpe, P. Khosla: A theoretical and experimental investigation of impact control for manipulators, *Int. J. Robot. Res.* **12**, 351–365 (1993)
- 7.64 J.K. Mills, D.M. Lokhorst: Control of robotic manipulators during general task execution: a discontinuous control approach, *Int. J. Robot. Res.* **12**, 146–163 (1993)
- 7.65 T.-J. Tarn, Y. Wu, N. Xi, A. Isidori: Force regulation and contact transition control, *IEEE Contr. Syst. Mag.* **16**(1), 32–40 (1996)
- 7.66 B. Brogliato, S. Niculescu, P. Orhant: On the control of finite dimensional mechanical systems with unilateral constraints, *IEEE Trans. Autom. Contr.* **42**, 200–215 (1997)

8. Robotic Systems Architectures and Programming

David Kortenkamp, Reid Simmons

Robot software systems tend to be complex. This complexity is due, in large part, to the need to control diverse sensors and actuators in real time, in the face of significant uncertainty and noise. Robot systems must work to achieve tasks while monitoring for, and reacting to, unexpected situations. Doing all this concurrently and asynchronously adds immensely to system complexity.

The use of a well-conceived architecture, together with programming tools that support the architecture, can often help to manage that complexity. Currently, there is no single architecture that is best for all applications – different architectures have different advantages and disadvantages. It is important to understand those strengths and weaknesses when choosing an architectural approach for a given application.

This chapter presents various approaches to architecting robotic systems. It starts by defining terms and setting the context, including a recounting of the historical developments in the area of robot architectures. The chapter then discusses in more depth the major types of architectural components in use today – behavioral control (Chap. 38), executives, and task planners (Chap. 9) – along with commonly used techniques for inter-

8.1 Overview	187
8.1.1 Special Needs of Robot Architectures	188
8.1.2 Modularity and Hierarchy	188
8.1.3 Software Development Tools	188
8.2 History	189
8.2.1 Subsumption	189
8.2.2 Layered Robot Control Architectures	190
8.3 Architectural Components	193
8.3.1 Connecting Components	193
8.3.2 Behavioral Control	195
8.3.3 Executive	196
8.3.4 Planning	199
8.4 Case Study – GRACE	200
8.5 The Art of Robot Architectures	202
8.6 Conclusions and Further Reading	203
References	204

connecting those components. Throughout, emphasis will be placed on programming tools and environments that support these architectures. A case study is then presented, followed by a brief discussion of further reading.

8.1 Overview

The term *robot architecture* is often used to refer to two related, but distinct, concepts. Architectural *structure* refers to how a system is divided into subsystems and how those subsystems interact. The structure of a robot system is often represented informally using traditional *boxes and arrows* diagrams or more formally using techniques such as unified modeling language (UML) [8.1]. In contrast, architectural *style* refers to the computational concepts that underlie a given system. For instance, one

robot system might use a publish–subscribe message passing style of communication, while another may use a more synchronous client–server approach.

All robotic systems use some architectural structure and style. However, in many existing robot systems it is difficult to pin down precisely the architecture being used. In fact, a single robot system will often use several styles together. In part, this is because the system implementation may not have clean subsystem bound-

aries, blurring the architectural structure. Similarly, the architecture and the domain-specific implementation are often intimately tied together, blurring the architectural style(s).

This is unfortunate, as a well-conceived, clean architecture can have significant advantages in the specification, execution, and validation of robot systems. In general, robot architectures facilitate development by providing beneficial constraints on the design and implementation of robotic systems, without being overly restrictive. For instance, separating behaviors into modular units helps to increase understandability and reusability, and can facilitate unit testing and validation.

8.1.1 Special Needs of Robot Architectures

In some sense, one may consider robot architectures as software engineering. However, robot architectures are distinguished from other software architectures because of the special needs of robot systems. The most important of these, from the architectural perspective, are that robot systems need to interact asynchronously, in real time, with an uncertain, often dynamic, environment. In addition, many robot systems need to respond at varying temporal scopes – from millisecond feedback control to minutes, or hours, for complex tasks.

To handle these requirements, many robot architectures include capabilities for acting in real time, controlling actuators and sensors, supporting concurrency, detecting and reacting to exceptional situations, dealing with uncertainty, and integrating high-level (symbolic) planning with low-level (numerical) control.

While the same capability can often be implemented using different architectural styles, there may be advantages of using one particular style over another. As an example, consider how a robot system's style of communications can impact on its reliability. Many robot systems are designed as asynchronous processes that communicate using message passing. One popular communication style is *client-server*, in which a message request from the client is paired with a response from the server. An alternate communication paradigm is *publish-subscribe*, in which messages are broadcast asynchronously and all modules that have previously indicated an interest in such messages receive a copy. With client-server-style message passing, modules typically send a request and then block, waiting for the response. If the response never comes (e.g., the server module crashes) then deadlock can occur. Even if the module does not block, the control flow still typically expects a response, which may lead to unexpected re-

sults if the response never arrives or if a response to some other request happens to arrive first. In contrast, systems that use publish-subscribe tend to be more reliable: because messages are assumed to arrive asynchronously, the control flow typically does not assume any particular order in which messages are processed, and so missing or out-of-order messages tend to have less impact.

8.1.2 Modularity and Hierarchy

One common feature of robot architectures is modular decomposition of systems into simpler, largely independent pieces. As mentioned above, robot systems are often designed as communicating processes, where the communications interface is typically small and relatively low bandwidth. This design enables the processes/modules to handle interactions with the environment asynchronously, while minimizing interactions with one another. Typically, this decreases overall system complexity and increases overall reliability.

Often, system decomposition is hierarchical – modular components are themselves built on top of other modular components. Architectures that explicitly support this type of layered decomposition reduce system complexity through abstraction. However, while hierarchical decomposition of robotic systems is generally regarded as a *desirable quality*, debate continues over the dimensions along which to decompose. Some architectures [8.2] decompose along a temporal dimension – each layer in the hierarchy operates at a characteristic frequency an order of magnitude slower than the layer below. In other architectures [8.3–6], the hierarchy is based on task abstraction – tasks at one layer are achieved by invoking a set of tasks at lower levels. In some situations, decomposition based on spatial abstraction may be more useful, such as when dealing with both local and global navigation [8.7]. The main point is that different applications need to decompose problems in different ways, and architectural styles can often be found to accommodate those different needs.

8.1.3 Software Development Tools

While significant benefit accrues from *designing* systems using well-defined architectural styles, many architectural styles also have associated software tools that facilitate adhering to that style during *implementation*. These tools can take the form of libraries of functions calls, specialized programming languages, or graphical editors. The tools make the constraints of the

architectural style explicit, while hiding the complexity of the underlying concepts.

For instance, inter-process communication libraries, such as common object request broker architecture (CORBA) [8.8] and inter-process communication (IPC) package [8.9], make it easy to implement message passing styles, such as client–server and publish–subscribe, respectively. Languages, such as Subsumption [8.10] and Skills [8.11] facilitate the development of data-driven, real-time behaviors, while languages such as the execution support language (ESL) [8.12] and the plan execution interchange language (PLEXIL) [8.13] provide support for reliably achieving higher-level tasks. Graphical editors, such as found in Control-Shell [8.14], Labview [8.15] and open robot controller

computer aided design (ORCCAD) [8.6], provide constrained ways to assemble systems, and then automatically generate code that adheres to the architectural style.

In each case, the tools facilitate the development of software in the given style and, more importantly, make it impossible (or, at least, very difficult) to violate the constraints of that architectural style. The result is that systems implemented using such tools are typically easier to implement, understand, debug, verify, and maintain. They also tend to be more reliable, since the tools provide well-engineered capabilities for commonly needed control constructs, such as message passing, interfacing with actuators and sensors, and handling concurrent tasks.

8.2 History

Robot architectures and programming began in the late 1960s with the Shakey robot at Stanford University [8.16] (Fig. 8.1). Shakey had a camera, a range finder, and bump sensors, and was connected to DEC PDP-10 and PDP-15 computers via radio and video links. Shakey’s architecture was decomposed into three



Fig. 8.1 Shakey (courtesy sri.com)



Fig. 8.2 The sense–plan–act (SPA) paradigm (after [8.3], with permission)

functional elements: sensing, planning, and executing [8.17]. The sensing system translated the camera image into an internal world model. The planner took the internal world model and a goal and generated a plan (i.e., a series of actions) that would achieve the goal. The executor took the plan and sent the actions to the robot. This approach has been called the *sense–plan–act* (SPA) paradigm (Fig. 8.2). Its main architectural features are that sensing flowed into a world model, which was then used by the planner, and that plan was executed without directly using the sensors that created the model. For many years, robotic control architectures and programming focused almost exclusively on the SPA paradigm.

8.2.1 Subsumption

In the early 1980s, it became apparent that the SPA paradigm had problems. First, planning in any real-world domain took a long time, and the robot would be blocked, waiting for planning to complete. Second, and more importantly, execution of a plan without involving sensing was dangerous in a dynamic world. Several new robot control architecture paradigms began to emerge, including reactive planning, in which plans were generated quickly and relied more directly on sensed information instead of internal models [8.4, 18]. The most influential work, however, was the subsumption architecture of Brooks [8.3]. A subsumption architecture is built from layers of interacting finite-state machines – each connecting sensors to actuators directly (Fig. 8.3). These finite-state machines were called

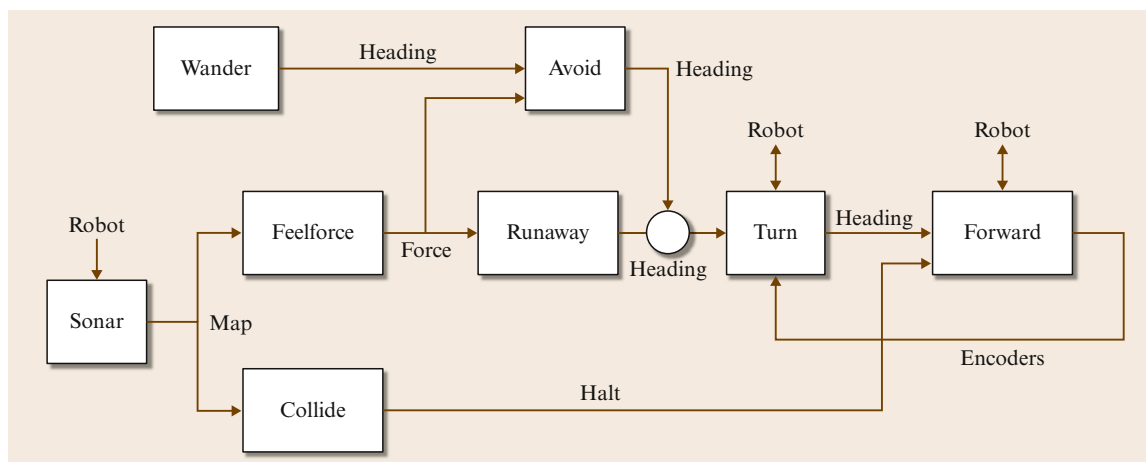


Fig. 8.3 Example of the Subsumption architecture (after [8.3], with permission)

behaviors (leading some to call Subsumption *behavior-based* or *behavioral* robotics [8.19]; see also Ch. 38). Since multiple behaviors could be active at any one time, Subsumption had an *arbitration* mechanism that enabled higher-level behaviors to override signals from lower-level behaviors. For example, the robot might have a behavior that simply drives the robot in random directions. This behavior is always active and the robot is always driving somewhere. A second, higher-level behavior could take sensor input, detect obstacles, and steer the robot away from them. It is also always active. In an environment with no obstacles, the higher-level behavior never generates a signal. However, if it detects an obstacle it overrides the lower-level behavior and steers the robot away. As soon as the obstacle is gone (and the higher-level behavior stops sending signals), the lower-level behavior gets control again. Multiple, interacting layers of behaviors could be built to produce more and more complex robots.

Many robots were built using the subsumption approach – most at MIT [8.20–22]. They were quite successful. Whereas SPA robots were slow and ponderous, Subsumption robots were fast and reactive. A dynamic world did not bother them because they constantly sensed the world and reacted to it. These robots scampered around like insects or small rodents. Several behavioral architectures arose in addition to Subsumption, often with different arbitration schemes for combining the outputs of behaviors [8.23, 24].

A popular example of behavior-based architectures is *Arkin's* motor-control schemas [8.25]. In this biologically inspired approach, motor and perceptual schemas [8.26] are dynamically connected to one an-

other. The motor schemas generate response vectors based on the outputs of the perceptual schemas, which are then combined in a manner similar to potential fields [8.27]. To achieve more complex tasks, the autonomous robot architecture (AuRA) architecture [8.28, 29] added a navigation planner and a plan sequencer, based on finite-state acceptors (FSAs), to the reactive schemas.

However, behavior-based robots soon reached limits in their capabilities. It proved very difficult to compose behaviors to achieve long-range goals and it proved almost impossible to optimize robot behavior. For example, a behavior-based robot that delivered mail in an office building could easily be built by simply wandering around the office building and having behaviors looking for rooms and then overriding the wandering and entering the office. It was much more difficult to use the behavioral style of architecture to design a system that reasoned about the day's mail to visit the offices in an optimal order to minimize delivery time. In essence, robots needed the planning capabilities of the early architectures wedded to the reactivity of the behavior-based architectures. This realization led to the development of layered, or tiered, robot control architectures.

8.2.2 Layered Robot Control Architectures

One of the first steps towards the integration of reactivity and deliberation was the reactive action packages (RAPs) system created by *Firby*. In his thesis [8.30], we see the first outline of an integrated, three-layer architecture. The middle layer of that architecture, and the subject of the thesis, was the RAPs system. *Firby* also

speculated on the form and function of the other two tiers, specifically with the idea of integrating classic deliberative methods with the ideas of the emerging situated reasoning community, but those layers were never implemented. Later, *Firby* would integrate *RAPs* with a continuous low-level control layer [8.31].

Independently and concurrently, *Bonasso* at MITRE [8.32] devised an architecture that began at the bottom layer with robot behaviors programmed in the Rex language as synchronous circuits [8.33]. These Rex machines guaranteed consistent semantics between the agent's internal states and those of the world. The middle layer was a conditional sequencer implemented in the Gapps language [8.34], which would continuously activate and deactivate the Rex skills until the robot's task was complete. This sequencer based on Gapps was appealing because it could be synthesized through more traditional planning techniques [8.35]. This work culminated in the 3T architecture (named after its three tiers of interacting control processes – planning, sequencing, and real-time control), which has been used on many generations of robots [8.36].

Architectures similar to 3T (Fig. 8.4) have been developed subsequently. One example is ATLANTIS [8.37], which leaves much more control at the sequencing tier. In ATLANTIS, the deliberative tier must be specifically called by the sequencing tier. A third example is *Saridis'* intelligent control architecture [8.38]. The architecture begins with the servo systems available on a given robot and augments them to integrate the execution algorithms of the next level, using VxWorks and the VME bus. The next level consists of a set of coordinating routines for each lower subsystem, e.g., vision, arm motion, and navigation. These are implemented in Petri net transducers (PNTs), a type of scheduling mechanism, and activated by a dispatcher connected to the organizational level. The organizational level is a planner implemented as a Boltzmann neural network. Essentially the neural network finds a sequence of actions that will match the required command received as text input, and then the dispatcher executes each of these steps via the network of PNT coordinators.

The LAAS architecture for autonomous systems (LAAS) is a three-layered architecture that includes software tools to support development/programming at each layer [8.39]. The lowest layer (functional) consists of a network of *modules*, which are dynamically parameterized control and perceptual algorithms. Modules are written in the generator of modules (*GenoM*) language, which produces standardized templates that facilitate the integration of modules with one another.

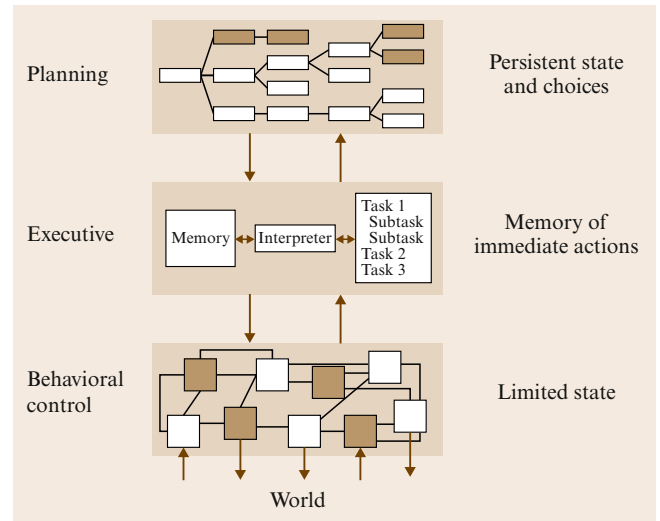


Fig. 8.4 Prototype three-tiered architecture

Unlike most other three-layered architectures, the executive layer is fairly simple – it is purely reactive and does no task decomposition. It serves mainly as a bridge – receiving task sequences from the highest layer and selecting and parameterizing tasks to send to the functional layer. The executive is written in the Kheops language, which automatically generates decision networks that can be formally verified. At the top, the decision layer consists of a planner, implemented using the indexed time table (*IxTeT*) temporal planner [8.40, 41], and a supervisor, implemented using procedural reasoning system (*PRS*) [8.42, 43]. The supervisor is similar to the executive layer of other three-layered architectures – it decomposes tasks, chooses alternative methods for achieving tasks, and monitors execution. By combining the planner and supervisor in one layer, *LAAS* achieves a tighter connection between the two, enabling more flexibility in when, and how, replanning occurs. The *LAAS* architecture actually allows for multiple decisional layers at increasingly higher levels of abstraction, such as a high-level *mission* layer and a lower-level *task* layer.

Remote agent is an architecture for the autonomous control of spacecraft [8.44]. It actually consists of four layers – a control (behavioral) layer, an executive, a planner/scheduler, and mode identification and recovery (*MIR*) that combines fault detection and recovery. The control layer is the traditional spacecraft real-time control system. The executive is the core of the architecture – it decomposes, selects, and monitors task execution, performs fault recovery, and does resource management,

turning devices on and off at appropriate times to conserve limited spacecraft power. The planner/scheduler is a batch process that takes goals, an initial (projected) state, and currently scheduled activities, and produces plans that include flexible ranges on start and end times

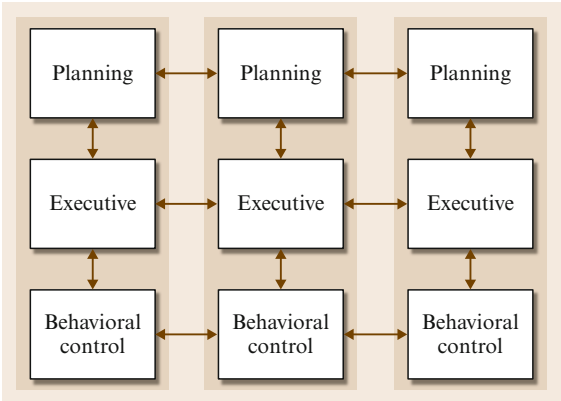


Fig. 8.5 The Syndicate multirobot architecture

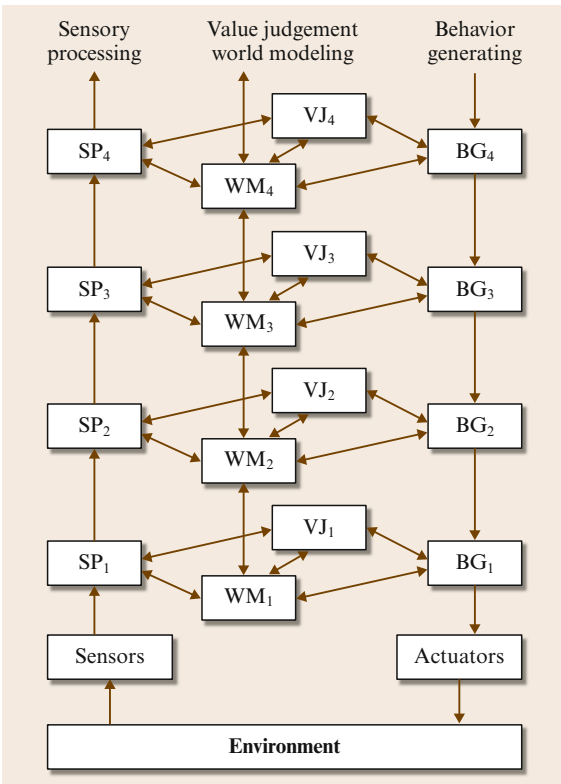


Fig. 8.6 The real-time control system (RCS) reference architecture (after [8.46], with permission)

of tasks. The plan also includes a task to reinvoke the planner to produce the next plan segment. An important part of the remote agent is configuration management – configuring hardware to support tasks and monitoring that the hardware remains in known, stable states. The role of configuration management is split between the executive, which uses reactive procedures, and **MIR**, which uses declarative models of the spacecraft and deliberative algorithms to determine how to reconfigure the hardware in response to detected faults [8.45].

The Syndicate architecture [8.47] extends the 3T model to multirobot coordination (Chap. 40). In this architecture, each layer interfaces not only with the layers above and below, as usual, but also with the layers of the other robots at the same level (Fig. 8.5). In this way, distributed control loops can be designed at multiple levels of abstraction. The version of Syndicate in [8.48] used a distributed market-based approach for task allocation at the planning layer.

Other noteworthy multitiered architectures have appeared in the literature. The National Bureau of Standards (NBS) developed for the Aeronautics and Space Agency (NASA) the NASA/NBS standard reference model (NASREM) [8.2, 49], later named real-time control system (RCS), was an early reference model for telerobotic control (Fig. 8.6). It is a many-tiered model in which each layer has the same general structure, but operates at increasingly lower frequency as it moves from the servo level to the reasoning levels. With the exception of maintaining a global world model, NASREM, in its original inception, provided for all the data and control paths that are present in architectures such as 3T, but NASREM was a reference model, not an implementation. The subsequent implementations of NASREM followed primarily the traditional sense–plan–act approach and were mainly applied to telerobotic applications, as opposed to autonomous robots. A notable exception was the early work of *Blidberg* [8.50].

While three-layered robot architectures are very popular, various two-layered architectures have been investigated by researchers. The coupled layered architecture for robot autonomy (CLARAty) was designed to provide reusable software for NASA’s space robots, especially planetary rovers [8.51, 52]. CLARAty consists of a functional and a decision layer. The functional layer is a hierarchy of object-oriented algorithms that provide more and more abstract interfaces to the robot, such as motor control, vehicle control, sensor-based navigation, and mobile manipulation. Each object provides a generic interface that is hardware independent, so that

the same algorithms can run on different hardware. The decision layer combines planning and executive capabilities. Similar to the **LAAS** architecture, this is done to provide for tighter coordination between planning and execution, enabling continual replanning in response to dynamic contingencies.

Closed-loop execution and recovery (**CLEaR**) [8.53] is one instantiation of the **CLARAty** decision layer. **CLEaR** combines the continuous activity scheduling, planning, execution and replanning (**CASPER**) repair-based planner [8.54] and the task description language (**TDL**) executive language [8.55]. **CLEaR** provides a tightly coupled approach to goal- and event-driven behavior. At its heart is the capability to do fast, continuous replanning, based on frequent state and resource updates from execution monitoring. This enables the planner to react to many exceptional situations, which can be important in cases where there are many tasks, few resources, and significant uncertainty. In **CLEaR**, both the planning and executive components are able to handle resource conflicts and exceptional situations – heuristics are used to decide which component should be involved in a given situation. The onboard autonomous science investigation system (**OASIS**) system [8.56] extends **CLEaR** to include science data analysis so that the architecture can be driven by opportunistic science-related goals (such as finding unusual rocks or formations). **OASIS** is planner-centric, releasing tasks to the executive component just a few seconds before their scheduled start times.

The cooperative intelligent real-time control architecture (**CIRCA**) is a two-layered architecture concerned with guaranteeing reliable behavior [8.57, 58]. It embodies the notion of *bounded reactivity* – an acknowl-

edgement that the resources of the robot are not always sufficient to guarantee that all tasks can be achieved. **CIRCA** consists of a real-time system (**RTS**) and an artificial intelligence (**AI**) system (**AIS**) that are largely independent. The **RTS** executes a cyclic schedule of test action pairs (**TAPs**) that have guaranteed worst-case behavior in terms of sensing the environment and conditionally acting in response. It is the responsibility of the **AIS** to create a schedule that is guaranteed to prevent catastrophic failures from occurring, while running in hard real time. The **AIS** does this by planning over a state-transition graph that includes transitions for actions, exogenous events, and the passage of time (e.g., if the robot waits too long, bad things can happen). The **AIS** tests each plan (set of **TAPs**) to see if it can actually be scheduled. If not, it alters the planning model, either by eliminating tasks (based on goal prioritization) or by changing parameters of behaviors (e.g., reducing the robot's velocity). The **AIS** continues this until it finds a plan that can be successfully scheduled, in which case it downloads the new plan to the **RTS** in an atomic operation.

Like **CIRCA**, **ORCCAD** is a two-layered architecture that is concerned with guaranteed reliability [8.6, 59]. In the case of **ORCCAD**, this guarantee is achieved through formal verification techniques. Robot tasks (lower-level behaviors) and robot procedures (higher-level actions) are defined in higher-level languages that are then translated into the Esterel programming language [8.60], for logical verification, or the Timed-Argus language [8.61], for temporal verification. The verification is geared toward liveness and safety properties, as well as verifying lack of contention for resources.

8.3 Architectural Components

We will take the three-tiered architecture as the prototype for the components discussed in this chapter. Figure 8.4 shows a typical three-tiered architecture. The lowest tier (or layer) is behavioral control and is the layer tied most closely to sensors and actuators. The second tier is the executive layer and is responsible for choosing the current behaviors of the robot to achieve a task. The highest tier is the task-planning layer and it is responsible for achieving long-term goals of the robot within resource constraints. Using the example of an office delivery robot, the behavioral layer is responsible for moving the robot around rooms and hallways,

for avoiding obstacles, for opening doors, etc. The executive layer coordinates the behavioral layer to achieve tasks such as leaving a room, going to an office, etc. The task-planning layer is responsible for deciding the order of deliveries to minimize time, taking into account delivery priorities, scheduling, recharging, etc. The task-planning layer sends tasks (e.g., exit the room, go to office 110) to the executive. All these tiers need to work together and exchange information. The next section deals with the problem of connecting components to each other. We then discuss each component of the three-tiered prototype architecture in detail.

8.3.1 Connecting Components

All of the architecture components that have been discussed in this chapter need to communicate with each other. They need to both exchange data and send commands. The choice of how components communicate (often called the *middleware*) is one of the most important and most constraining of the many decisions a robot architecture designer will make. From previous experience, a great deal of the problems and a majority of the debugging time in developing robot architectures have to do with communication between components. In addition, once a communication mechanism is chosen it becomes extremely difficult to change, so early decisions persist for many years. Many developers *roll their own* communication protocols, usually built on top of Unix sockets. While this allows for customization of messages, it fails to take advantage of the reliability, efficiency, and ease of use that externally available communication packages provide. There are two basic approaches to communication: client–server and publish–subscribe.

Client–Server

In a client–server (also called a point-to-point) communication protocol, components talk directly with other components. A good example of this is remote procedure call (RPC) protocols in which one component (the client) can call functions and procedures of another component (the server). A modern, and popular, variation on this is the common object request broker architecture (CORBA). CORBA allows for one component to call object methods that are implemented by another component. All method calls are defined in an interface definition language (IDL) file that is language independent. Every component uses the same IDL to generate code that compiles with component to handle communication. The advantage of this is that, when an IDL file is changed, all components that use that IDL can be recompiled automatically (by using *make* or similar code configuration tools). CORBA object request brokers (ORBs) are available for most major object-oriented languages. Although free ORBs are available, many commercial ORBs offer additional features and support. One disadvantage of CORBA is that it introduces quite a bit of additional code into applications. Some competitors have tried to address this issue, such as the internet communications engine (ICE), which has its own version of an IDL file called the specification language for ICE (SLICE). The biggest advantage of a client–server protocol is that the interfaces are very clearly

defined in advance and everyone knows when the interface has changed. Another advantage is that it allows for a distributed approach to communication with no central module that must distribute data. A disadvantage of client–server protocols is that they introduce significant overhead, especially if many components need the same information. It should be noted that CORBA and ICE also have a broadcast mechanism (called an event channel, or the notification service, in CORBA).

Publish–Subscribe

In a publish–subscribe (also called a broadcast) protocol, a component publishes data and any other component can subscribe to that data. Typically, a centralized process routes data between publishers and subscribers. In a typical architecture, most components both publish information and subscribe to information published by other components. There are several existing publish–subscribe middleware solutions. A popular one for robotics is the real-time innovations’ (RTI) data distribution service (DDS), formerly the network data distribution service (NDDS) [8.62]. Another popular publish–subscribe paradigm is IPC developed at Carnegie Mellon University [8.9]. Many publish–subscribe protocols are converging on using extensible markup language (XML) descriptions to define the data being published, with the added convenience of transmitting XML over HTTP, which allows for significant interoperability with Web-based applications. Publish–subscribe protocols have a large advantage in being simple to use and having low overhead. They are especially useful when it is unknown how many different components might need a piece of data (e.g., multiple user interfaces). Also, components do not get bogged down with repeated requests for information from many different sources. Publish–subscribe protocols are often more difficult to debug because the syntax of the message is often hidden in a simple string type. Thus problems are not revealed until runtime when a component tries, and fails, to parse an incoming message. Publish–subscribe protocols are also not as readable when it comes to sending commands from one module to another. Instead of calling an explicit method or function with parameters, a command is issued by publishing a message with the command and parameters in it and then having that message be parsed by a subscriber. Finally, publish–subscribe protocols often use a single central server to dispatch messages to all subscribers, providing a single point of failure and potential bottleneck.

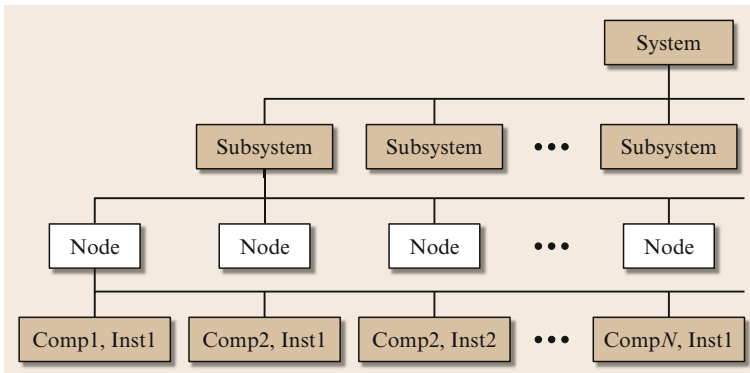


Fig. 8.7 The JAUS reference architecture topology (after JAUS Reference Architecture document [8.63])

JAUS

Recently, a standard has emerged in the defense robotics community not only for a communication protocol but also for definitions of messages that are sent via that communication protocol. The joint architecture for unmanned systems (JAUS) defines a set of reusable messages and interfaces that can be used to command autonomous systems [8.63–65]. These reusable components reduce the cost of integrating new hardware components into autonomous systems. Reuse also allows for components developed for one autonomous system to be used by another autonomous system. JAUS has two components: a *domain model* and a *reference architecture*. The domain model is a representation of the unmanned systems' functions and information. It contains a description of the system's functional and informational capabilities. The former includes models of the system's maneuvering, navigational, sensing, payload, and manipulation capabilities. The latter includes models of the system's internal data, such as maps and system status. The reference architecture provides a well-defined set of messages. Messages cause actions to commence, information to be exchanged, and events to occur. Everything that occurs in a JAUS system is precipitated by messages. This strategy makes JAUS a component-based, message-passing architecture.

The JAUS reference architecture defines a system hierarchy, as shown in Fig. 8.7. The topology defines the *system* as the collection of vehicles, operator control units (OCU), and infrastructure necessary to provide the full robotic capability. Subsystems are individual units (e.g., vehicles or OCUs) in the system. Nodes define a distinct processing capability within the architecture and route JAUS messages to components. Components provide the different execution capabilities and respond directly to command messages. Components might be

sensors (e.g., a SICK laser or a vision sensor), actuators (a manipulator or a mobile base) or payloads (weapons or task sensors). The topology (the layout of particular system, subsystems, nodes, and components) is defined by the system implementers based on task requirements.

At the core of JAUS is a set of well-defined messages. JAUS supports the following message types.

<i>Command:</i>	Initiate mode changes or actions
<i>Query:</i>	Used to solicit information from a component
<i>Inform:</i>	Response to a query
<i>Event set up:</i>	Passes parameters to set up an event
<i>Event notification:</i>	Sent when the event happens

JAUS has about 30 predefined messages that can be used to control robots. There are messages for control of a robotic vehicle. For example, the *global vector driver* message performs closed-loop control of the desired global heading, altitude, and speed of a mobile vehicle. There are also sensor messages such as *global pose sensor*, which distributes the global position and orientation of the vehicle. There are also manipulation messages in JAUS. For example, the *set joint positions* message sets the desired joint position values. The *set tool point* message specifies the coordinates of the end-effector tool point in terms of the coordinate system attached to the end-effector.

JAUS also has user-definable messages. Messages have headers that follow a specific format and include message type, destination address (e.g., system, subsystem, node, and component), priority, etc. While JAUS is primarily point to point, JAUS messages can also be marked as *broadcast* and distributed to all components. JAUS also defines coordinate systems for navigation and manipulation to ensure all components understand any coordinates sent to them.

8.3.2 Behavioral Control

Behavioral control represents the lowest level of control in a robot architecture. It directly connects sensors and actuators. While these are typically hand-crafted functions written in C or C++, there have been specialized languages developed for behavioral control, including ALFA [8.66], Behavioral Language [8.67], and Rex [8.68]. It is at this level that traditional control theory (e.g., PID functions, Kalman filters, etc.) resides. In architectures such as 3T, the behavioral layer functions as a *Brooksonian machine* – that is, the layer is composed of a small number of behaviors (also called skills) that perceive the environment and carry out the actions of the robot.

Example

Consider an office delivery robot that operates in a typical office building. The behavioral control layer contains the control functions necessary to move around in the building and carry out delivery tasks. Assuming the robot has an *a priori* map of the building, some possible behaviors for this robot include

1. move to location while avoiding obstacles
2. move down hallway while avoiding obstacles
3. find a door
4. find a door knob
5. grasp a door knob
6. turn a door knob
7. go through door
8. determine location
9. find office number
10. announce delivery

Each of these behaviors ties sensors (vision, range sensing, etc.) to actuators (wheel motors, manipulator motors, etc.) in a tight loop. In architectures such as Subsumption, all behaviors are running concurrently with a hierarchical control scheme inhibiting the outputs of certain behaviors. In AuRA [8.29], behaviors are combined using potential functions. Other architectures [8.24, 68] use explicit *arbitration* mechanisms to choose amongst potentially conflicting behaviors.

In architectures such as 3T [8.36], not all of the behaviors are active at the same time. Typically, only a few behaviors that do not conflict would be active at a time (e.g., behaviors 2 and 9 in the example above). The executive layer (see Sect. 8.3.3) is responsible for activating and deactivating behaviors to achieve higher-level tasks and to avoid conflicts between two behaviors competing for the same resource (e.g., an actuator).

Situated Behaviors

An important aspect of these behaviors is that they be *situated*. This means that the behavior works only in very specific situations. For example, behavior 2 above moves down a hallway, but this is appropriate only when the robot is situated in a hallway. Similarly, behavior 5, which grasps a door knob, is appropriate only when the robot is within grasping distance of a door knob. The behavior is not responsible for putting the robot in the particular situation. However, it should recognize that the situation is not appropriate and signal as such.

Cognizant Failure

A key requirement for behaviors is that they know when they are not working. This is called *cognizant failure* [8.69]. For example, behavior 5 in our example (grasping the door knob) should not continually grasp at air if it is failing. More succinctly, the behavior should not continue to *bang its head against the wall*. A common problem with early Subsumption robots is that the behaviors did not know they were failing and continued to take actions that were not resulting in progress. It is not the job of the behavioral control layer to decide what to do in a failure situation; it is only necessary to announce that the behavior has failed and halt activity.

Implementation Constraints

The behavioral control layer is designed to bring the speed and reactivity of Subsumption to robot control. For this reason, the behaviors in the behavioral control layer need to follow the philosophies of Subsumption. In particular, the algorithms used for behaviors should be constant in state and time complexity. There should be little or no search at the behavioral control level, and little iteration. Behaviors should simply be transfer functions that take in signals (from sensors or other behaviors) and send out signals (to actuators or other behaviors), and repeat these several times a second. This will allow for reactivity to changing environments. More controversial is how much state should be allowed at the behavioral level. Brooks famously said several years ago to “use the world as its own best model” [8.67] – that is, instead of maintaining internal models of the world and querying those models, the robot should instead directly sense the world to get its data. State such as maps, models, etc. belong at the higher levels of the three-tiered prototype architecture, not at the behavioral control layer. Certain exceptions, such as maintaining state for data filtering calculations, could be made on a case-by-case basis. Gat [8.70] argues that any state kept at the behavioral layer should be ephemeral and limited.

8.3.3 Executive

The executive layer is the interface between the numerical behavioral control and the symbolic planning layers. The executive is responsible for translating high-level plans into low-level behaviors, invoking behaviors at the appropriate times, monitoring execution, and handling exceptions. Some executives also allocate and monitor resource usage, although that functionality is more commonly performed by the planning layer.

Example

Continuing the example of an office delivery robot, the main high-level task would be to deliver mail to a given office. The executive would decompose this task into a set of subtasks. It may use a geometric path planner to determine the sequence of corridors to move down and intersections at which to turn. If there are doorways along the route, a task would be inserted to open and pass through the door. At the last corridor, the executive would add a concurrent task that looks for the office number. The final subtasks would be to announce that the person has mail and to concurrently monitor whether the mail has been picked up. If it is not picked up after some period of time, an exception would be triggered that invokes some recovery action (perhaps announcing again, perhaps checking to make sure the robot is at the correct office, perhaps notifying the planning layer to reschedule the delivery for a later time).

Capabilities

The example above illustrates many of the capabilities of the executive layer. First, the executive decomposes high-level tasks (goals) into low-level tasks (behaviors). This is typically done in a *procedural* fashion: the knowledge encoded in the executive describes *how* to achieve tasks, rather than describing *what* needs to be done and having the executive figure out the *how* by itself. Sometimes, though, the executive may also use specialized planning techniques, such as the route planner used in the example above. The decomposition is typically a hierarchical *task tree* (Fig. 8.8), with the leaves of the task tree being invocations and parameterizations of behaviors.

Besides decomposing tasks into subtasks, executives add and maintain temporal constraints between tasks (usually between sibling tasks only, but some executive languages permit temporal constraints between any pair of tasks). The most common constraints are *serial* and *concurrent*, but most executives support more expressive constraint languages, such as having one task begin 10 s

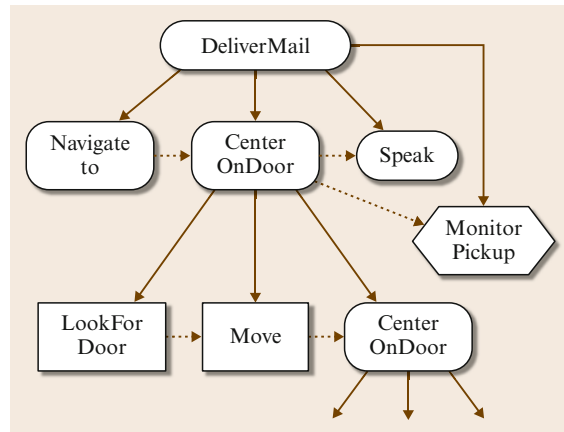


Fig. 8.8 Hierarchical task tree for mail-delivery task (lozenge nodes are interior; rectangular nodes are leaves; hexagonal node is an execution monitor; solid arrows are parent-child relationships; dashed arrows are sequential constraints)

after another one starts or having one task end when another ends.

The executive is responsible for dispatching tasks when their temporal constraints are satisfied. In some executives, tasks may also specify resources (e.g., the robot's motors or camera) that must be available before the task can be dispatched. As with behaviors, arbitrating between conflicting tasks can be a problem. In the case of executives, however, this arbitration is typically either programmed in explicitly (e.g., a rule that says what to do in cases where the robot's attempt to avoid obstacles takes it off the preferred route) or handled using priorities (e.g., recharging is more important than mail delivery).

The final two important executive capabilities are execution monitoring and error recovery. One may wonder why these capabilities are needed if the underlying behaviors are reliable. There are two reasons. First, as described in Sect. 8.3.2, the behaviors are situated, and the situation may change unexpectedly. For instance, a behavior may be implemented assuming that a person is available to pick up the mail, but that may not always be the case. Second, in trying to achieve some goal, the behavior may move the robot into a state that is unexpected by the executive. For instance, people may take advantage of the robot's obstacle avoidance behavior to *herd* it into a closet. While the behavior layer may, in fact, keep the robot safe in such situations, the executive needs to detect the situation in order to get the robot back on track.

Typically, execution monitoring is implemented as a concurrent task that either analyzes sensor data directly or activates a behavior that sends a signal to the executive when the monitored situation arises. These correspond to *polling* and *interrupt-driven* monitors, respectively.

Executives support various responses to monitors being triggered. A monitor may spawn subtasks that handle the situation, it may terminate already spawned subtasks, it may cause the parent task to fail, or it may raise an exception. The latter two responses involve the *error recovery* (also called *exception handling*) capability. Many executives have tasks return status values (success or failure) and allow parent tasks to execute conditionally based on the return values. Other executives use a hierarchical exception mechanism that throws named exceptions to ancestor nodes in the task tree. The closest task that has registered a handler for that exception tries to handle it; if it cannot, it rethrows the exception up the tree. This mechanism, which is inspired by the exception handling mechanisms of C++, Java, and Lisp, is strictly more expressive than the return-value mechanism, but it is also much more difficult to design systems using that approach, due to the nonlocal nature of the control flow.

Implementation Constraints

The underlying formalism for most executives is a hierarchical finite-state controller. Petri nets [8.71] are a popular choice for representing executive functions. In addition, various languages have been developed specifically to assist programmers in implementing executive-level capabilities. We briefly discuss aspects of several of these languages: reactive action packages (RAPs) [8.4, 30], the procedural reasoning system (PRS) [8.42, 43], the execution support language (ESL) [8.12], the task description language (TDL) [8.55], and the plan execution interchange language (PLEXIL) [8.13].

These languages all share features and exhibit differences. One distinction is whether the language is stand-alone (RAPs, PRS, PLEXIL) or an extension of an existing language (ESL is an extension of Common Lisp; TDL is an extension of C++). Stand-alone languages are typically easier to analyze and verify, but extensions are more flexible, especially with respect to integration with legacy software. While stand-alone executive languages all support interfaces to user-defined functions. These interfaces are usually limited in capability (such as what types of data structures can be passed around).

All of these executive languages provide support for hierarchical decomposition of tasks into subtasks. All

except PLEXIL allow for recursive invocation of tasks. RAPs, TDL, and PLEXIL have syntax that distinguishes leaf nodes of the task tree/graph from interior nodes.

All these languages provide capabilities for expressing conditionals and iteration, although with RAPs and PLEXIL these are not core-language constructs, but must be expressed as combinations of other constructs. Except for TDL, the languages all provide explicit support for encoding pre- and post-conditions of the tasks and for specifying success criteria. With TDL, these concepts must be programmed in, using more primitive constructs. The stand-alone languages all enable local variables to be defined within a task description, but provide for only limited computation with those variables. Obviously, with extension languages the full capability of the base language is available for defining tasks.

All the languages support the simple serial (sequential) and concurrent (parallel) temporal constraints between tasks, as well as timeouts that can be specified to trigger after waiting a specified amount of time. In addition, TDL directly supports a wide range of temporal constraints – one can specify constraints between the start and end times of tasks (e.g., *task B starts after task A starts* or *task C ends after task D starts*) as well as metric constraints (e.g., *task B starts 10 seconds after task A ends* or *task C starts at 1pm*). ESL and PLEXIL support the signaling of events (e.g., when tasks transition to new states) that can be used to implement similarly expressive types of constraints. In addition, ESL and TDL support task termination based on the occurrence of events (e.g., *task B terminates when task A starts*).

The languages presented differ considerably in how they deal with execution monitoring and exception handling. ESL and TDL both provide explicit execution monitoring constructs and support exceptions that are *thrown* and then *caught* by registered handlers in a hierarchical fashion. This type of exception handling is similar to that used in C++, Java, and Lisp. ESL and TDL also support *clean-up* procedures that can be invoked when tasks are terminated. RAPs and PLEXIL use return values to signal failure, and do not have hierarchical exception handling. PLEXIL, though, does support clean up procedures that are run when tasks fail. PRS has support for execution monitoring, but not exception handling. ESL and PRS support the notion of *resources* that can be shared. Both provide support for automatically preventing contention amongst tasks for the resources. In the other executive languages, this must be implemented separately (although there are plans to extend PLEXIL in this area).

Finally, **RAPs**, **PRS** and **ESL** all include a symbolic database (*world model*) that connects either directly to sensors or to the behavior layer to maintain synchrony with the real world. Queries to the database are used to determine the truth of preconditions, to determine which methods are applicable, etc. PLEXIL has the concept of a *lookup* that performs a similar function, although it is transparent to the task how this is implemented (e.g., by a database lookup or by invoking a behavior-level function). **TDL** leaves it up to the programmer to specify how the tasks connect to the world.

8.3.4 Planning

The planning component of our prototype layered architecture is responsible for determining the long-range activities of the robot based on high-level goals. Where the behavioral control component is concerned with the here-and-now and the executive is concerned with what has just happened and what should happen next, the planning component looks towards the future. In our running example of an office delivery robot, the planning component would look at the day's deliveries, the resources of the robot, and a map, and determine the optimal delivery routes and schedule, including when the robot should recharge. The planning component is also responsible for replanning when the situation changes. For example, if an office is locked, the planning component would determine a new delivery schedule that puts that office's delivery later in the day.

Types of Planning

Chapter 9 describes approaches to robot planning in detail. Here, we summarize issues with respect to different types of planners as they relate to layered architectures.

The two most common approaches used are hierarchical task net (**HTN**) planners and planner/schedulers. **HTN** planners [8.72, 73] decompose tasks into subtasks, in a manner similar to what many executives do. The main differences are that **HTN** planners typically operate at higher levels of abstraction, take resource utilization into account, and have methods for dealing with conflicts between tasks (e.g., tasks needing the same resources, or one task negating a precondition needed by another task). The knowledge needed by **HTN** planners is typically fairly easy to specify, since one indicates directly *how* tasks are to be achieved.

Planner/schedulers [8.74, 75] are useful in domains where time and resources are limited. They create high-level plans that schedule when tasks should occur, but typically leave it to the executive to determine exactly

how to achieve the tasks. Planner/schedulers typically work by laying out tasks on time lines, with separate time lines for the various resources that are available on the robot (motors, power, communication, etc.). The knowledge needed by planner/schedulers includes the goals that tasks achieve, the resources they need, their duration, and any constraints between tasks.

Many architectures provide for specialized planning *experts* that are capable of solving particular problems efficiently. In particular, these include motion planners, such as path planners and trajectory planners. Sometimes, the planning layer of the architecture invokes these specialized planners directly; in other architectural styles, the motion planners are part of the lower levels of the architecture (the executive, or even the behavioral layer). Where to put these specialized planners is often a question of style and performance (see Sect. 8.5).

Additionally, some architectures provide for multiple planning layers [8.39, 44, 76]. Often, there is a *mission* planning layer at the very top that plans at a very abstract level, over relatively long periods of time. This layer is responsible mainly for selecting which high-level goals are to be achieved over the next period of time (and, in some cases, determining in which order to achieve them) in order to maximize some objective function (e.g., net reward). The lower *task* planning layer is then responsible for determining exactly how and when to achieve each goal. This breakdown is usually done for efficiency reasons, since it is difficult to plan simultaneously at both a detailed level and over a long time horizon.

Integrating Planning and Execution

There are two main approaches to the integration of the planning and execution components in robotic architectures. The first approach is that the planning component is invoked as needed by the executive and returns a plan. The planning component is then dormant until called again. Architectures such as ATLANTIS [8.70] and Remote Agent [8.44] use this approach, which requires that the executive either leave enough time for planning to complete or that it *saves* the system until planning is complete. In the Remote Agent, for instance, a special *planning* task is explicitly scheduled.

The second approach is that the planning component sends high-level tasks down to the executive as required and monitors the progress of those tasks. If tasks fail, replanning is done immediately. In this approach, the planning component is always running and always planning and replanning. Signals must pass in real time between the planner and the executive to keep

them synchronized. Architectures such as 3T [8.36] use this second approach. The first approach is useful when the system is relatively static, so that planning can occur infrequently, at relatively predictable times. The second approach is more suited to dynamic environments, where replanning is more frequent and less predictable.

Other decisions that need to be made when integrating planning and execution are when to stop task decomposition, where to monitor plan execution, and how to handle exceptions. By planning all the way down to primitive actions/behaviors, the planner has a very good notion of what will happen during execution, but at a price of much more computation. Also, some task decompositions are easier to describe procedurally (using an executive language) rather than declaratively (using a planning language). Similarly, monitoring at the execu-

tive level tends to be more efficient, since the monitoring happens closer to the robot sensors, but the planner may be able to use its more global knowledge to detect exceptions earlier and/or more accurately. With respect to handling exceptions, executives can handle many on their own, at the price of breaking the expectations used by the planner in scheduling tasks. On the other hand, having exceptions handled by the planner typically involves replanning, which can be computationally expensive.

For all these integration issues, however, a middle ground usually exists. For instance, one can choose to decompose some tasks more deeply than others, or handle certain exceptions in the executive and others in the planner. In general, the *right* approach usually involves a compromise and is determined by analyzing the domain in detail (see Sect. 8.5).

8.4 Case Study – GRACE

In this section, we present the architecture of a fairly complex autonomous mobile robot. Graduate robot attending conference (**GRACE**) resulted from the efforts of five research institutions (Carnegie Mellon, Naval Research Laboratory, Northwestern University, Metricka, and Swarthmore College) to tackle the American Association for Artificial Intelligence (**AAAI**) Robot Challenge. The Challenge was for a robot to attend the **AAAI** National Conference on Artificial Intelligence as a participant – the robot must find the registration desk (without knowing the layout of the convention center beforehand), register for the conference, and then, after being provided with a map, find its way to a given location in time to give a technical talk about itself.

The architectural design of the robot was particularly important given the complexity of the task and the need to integrate techniques that had been previously developed by the five institutions. These techniques included localization in a dynamic environment, safe navigation in the presence of moving people, path planning, dynamic replanning, visual tracking of people, signs and landmarks, gesture and face recognition, speech recognition and natural language understanding, speech generation, knowledge representation, and social interaction with people.

GRACE is built on top of an real world interface (**RWI**) B21 base and has an expressive computer-animated face projected on a flat-panel liquid-crystal display (**LCD**) screen (Fig. 8.9). Sensors that come with the B21 include touch, infrared, and sonar sen-

sors. Near the base is a **SICK** scanning laser range finder that provides a 180° field of view. In addition, **GRACE** has several cameras, including a stereo camera head on a pan-tilt unit (**PTU**) built by Metricka **TRAC**Labs and a single-color camera with pan-tilt-zoom capability, built by Canon. **GRACE** can speak using a high-quality speech-generation software (**Festival**), and receive speech responses using a wireless microphone headset (a **Shure TC** computer wireless transmitter/receiver pair).

The behavioral layer of the architecture consisted of individual processes that controlled particular pieces of hardware. These programs provided abstract interfaces to either control the hardware or return information from



Fig. 8.9 The robot **GRACE**

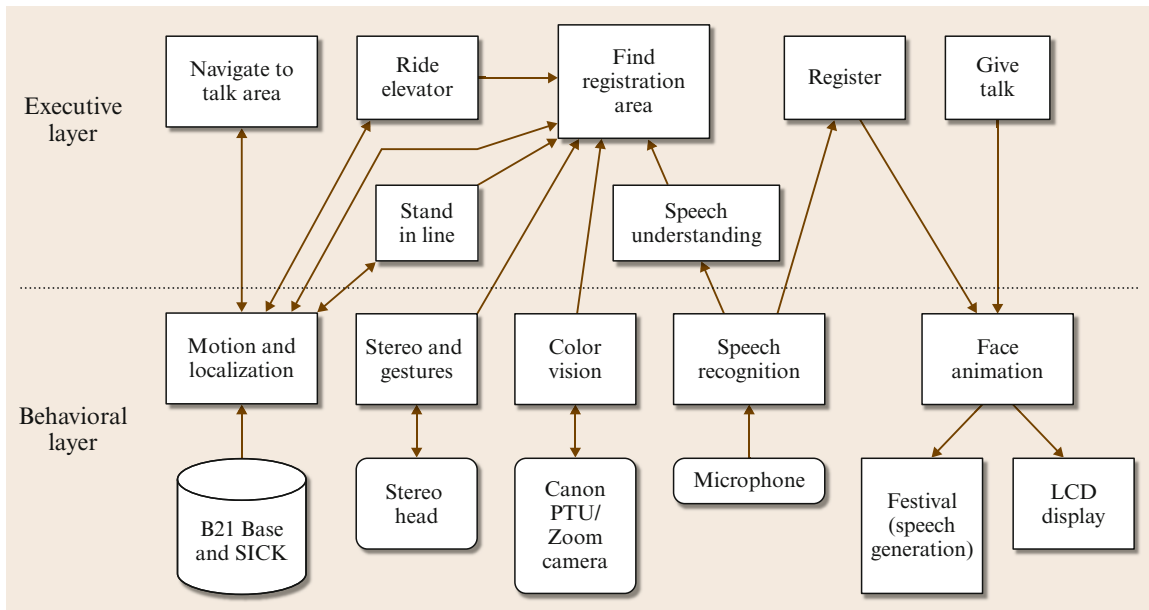


Fig. 8.10 GRACE's architectural structure

sensors. To accommodate the different coding styles of the various groups involved, both synchronous, blocking and asynchronous, nonblocking calls were supported by most of the interfaces (for the nonblocking calls, the interfaces allowed programmers to specify a callback function to be invoked when data was returned). Interfaces at the behavioral level included robot motion and localization (this interface also provided laser information), speech recognition, speech generation, facial animation, color vision, and stereo vision (Fig. 8.10).

The architecture used individual processes for each of the behavioral capabilities, mainly because the underlying code had been developed by different organizations. While having a large number of processes run concurrently is somewhat inefficient, trying to integrate everything into a monolithic process was thought to be too difficult. In addition, the use of separate processes facilitated development and debugging, since one needed to run only those aspects of the system that were being tested.

The executive layer consisted of separate programs for achieving each subtask of the challenge – finding the registration desk, riding the elevator, standing in line, interacting with the person at the desk, navigating to the talk, and giving the talk (Fig. 8.10). As is common in many implemented robotic systems, the GRACE architecture did not have a planning layer – since the high-level plan was fixed and relatively straightforward,

it was coded explicitly. Several of the executive-layer programs were written using TDL (see Sect. 8.3.3),

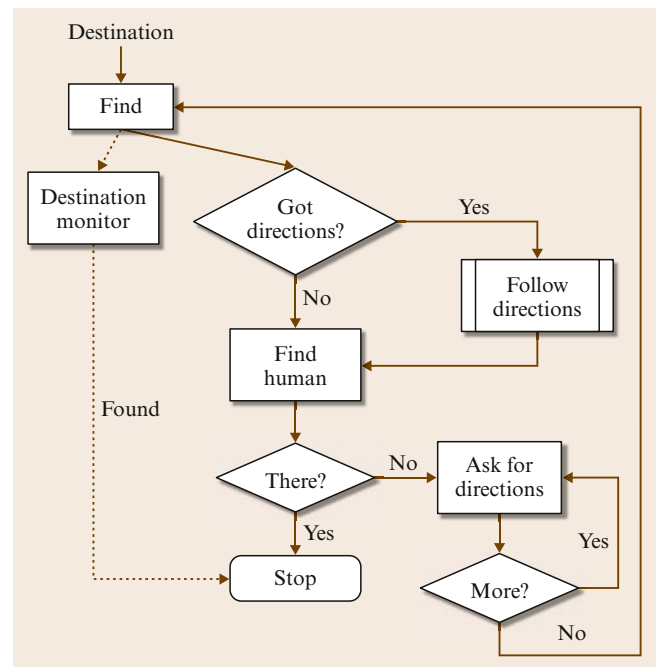


Fig. 8.11 Finite-state machine for GRACE's task for following directions to the registration booth

which facilitated concurrent control and monitoring of the various tasks.

One particularly involved task was finding the registration desk (recall that **GRACE** had no idea where the booth was, or even what the convention center looked like). **TDL** was used to create a finite-state machine that allowed **GRACE** to maintain multiple goals, such as using an elevator to get to a particular floor and following directions to find the elevator (Fig. 8.11). The top-level goal was to find the registration desk. Intermediate subgoals were created as **GRACE** interacted with people to determine the directions to the desk. If there were no directions to follow, **GRACE** performed a random walk until a person was detected using its laser scanner. **GRACE** then engaged in conversation with the person to obtain directions. **GRACE** could handle simple commands, such as *turn left* and *go forward five meters*, as well as higher-level instructions, such as *take the elevator* and *turn left at the next intersection*. In addition, **GRACE** could ask questions, such as *am I at the registration desk?* and *is this the elevator?* The **TDL**-based finite-state machine was used to determine which interactions were appropriate at various times and to prevent the robot from getting confused.

Communication between processes used the inter-process communication (**IPC**) messaging package [8.9, 77]. **IPC** supports both publish-subscribe and client-

server messaging, and enables complex data structures to be passed transparently between processes. One side benefit of using **IPC** to communicate between processes was the ability to log all message traffic (both message name and data content). This proved invaluable, at times, in determining why the system failed to act as expected – did a process send out a message with invalid data? Did it fail to send out a message in a timely fashion? Was the receiving process blocked, for some reason? Was there a timing issue? While wading through the message traffic was often tedious, in some cases it was the only way to catch intermittent bugs.

In July 2002, **GRACE** successfully completed the challenge at the Shaw Convention Centre in Edmonton, Canada. The processes at the behavioral level generally worked as anticipated – this was largely attributed to the fact that those modules were ported from previously developed (and hence well-tested) systems. While generally functional, the executive-level processes had more problems with off-nominal situations. This is largely attributed to problems in sensor interpretation, as well as mistaken assumptions about what the convention center was going to look like (for instance, it turned out that some of the barriers were made of glass, which is largely invisible to the laser). Overall, however, the architecture itself worked as expected, enabling a large body of complex software to be integrated rather quickly and operate together effectively.

8.5 The Art of Robot Architectures

Designing a robot architecture is much more of an art than a science. The goal of an architecture is to make programming a robot easier, safer, and more flexible. Thus, the decisions made by a developer of a robot architecture are influenced by their own prior experiences (e.g., what programming languages they are familiar with), their robot and its environment, and the tasks that need to be performed. The choice of a robot architecture should not be taken lightly, as it is the authors' experiences that early architectural decisions often persist for years. Changing robot architectures is a difficult proposition and can set back progress while a great deal of code is reimplemented.

The art of designing a robotic architecture starts with a set of questions that the designer needs to ask. These questions include:

- What are the tasks the robot will be performing? Are they long-term tasks? Short-term? User-initiated?

Robot-initiated? Are the tasks repetitive or different across time?

- What actions are necessary to perform the tasks? How are those actions represented? How are those actions coordinated? How fast do actions need to be selected/changed? At what speed do each of the actions need to run in order to keep the robot safe?
- What data is necessary to do the tasks? How will the robot obtain that data from the environment or from the users? What sensors will produce the data? What representations will be used for the data? What processes will abstract the sensory data into representations internal to the architecture? How often does the data need to be updated? How often can it be updated?
- What computational capabilities will the robot have? What data will these computational capabilities produce? What data will they consume? How will

the computational capabilities of a robot be divided, structured, and interconnected? What is the best decomposition/granularity of computational capabilities? How much does each computational capability have to know about the other capabilities? Are there legacy computational capabilities (from other robots, other robot projects, etc.) that will be used? Where will the different computational capabilities reside (e.g., onboard or offboard)?

- Who are the robot's users? What will they command the robot to do? What information will they want to see from the robot? What understanding do they need of the robot's computational capabilities? How will the user know what the robot is doing? Is the user interaction peer to peer, supervisory, or as a bystander?
- How will the robot be evaluated? What are the success criteria? What are the failure modes? What is the mitigation for those failure modes?
- Will the robot architecture be used for more than one set of tasks? For more than one kind of robot? By more than one team of developers?

Once designers have answers to all (or most) of these questions, they can then begin building some *use cases* for the types of operations they want the robot to perform and how they want users to interact with it. These use cases should specify the outward behavior of the robot with respect to its environment and its users. From the use cases, an initial partitioning of robot functionality can be developed. This partitioning should be accompanied by a *sequence diagram* that shows the transfer of information and control over time amongst the various components of the robot architecture [8.78]. After this, a more formal specification of the interfaces

between architectural components can be developed. This may be done using a language such as the interface definition language (IDL) of CORBA or by defining the messages to be distributed in a publish–subscribe protocol. This is an important step, as once implementation begins it is very costly to change interfaces. If an interface does change, all stakeholders need to be notified and need to agree to the change. The most common integration problems in robot architectures are mismatches between what components expect and what they are receiving in the way of data.

An advantage of tiered architectures with clear interface definitions is that the different layers can be developed in parallel. The behavioral control layer can be implemented and tested on the robot using a human as an executive. The executive can be implemented and tested using state machine *stubs* for the expected behaviors on the robot. The planning layer can be implemented and tested using *stubs* for the tasks in the executive. The *stubs* merely acknowledge that they were called and report back appropriately. Then, the tiers can be integrated to test timing and other runtime issues. This parallel approach speeds up the development of a robot architecture, but is possible only if the roles and interfaces between components are clearly defined and respected. There is still considerable real-time debugging necessary during integration. In our experiences, most of the development time in robot architectures is still spent on the behavioral control layer – that is, sensing and acting are still the hard parts of robot control, as compared to execution and planning. Having a good, robust behavioral control layer goes a long way towards having a competent robot architecture.

8.6 Conclusions and Further Reading

Robot architectures are designed to facilitate the concurrent execution of task-achieving behaviors. They enable systems to control actuators, interpret sensors, plan, monitor execution, and deal with unexpected contingencies and opportunities. They provide the conceptual framework within which domain-dependent software development can take place, and they often provide programming tools that facilitate that development.

While no single architecture has proven to be best for all applications, researchers have developed a variety of approaches that can be applied in different

situations. While there is not yet a specific formula for determining which architecture will be best suited for a given application, this chapter provides some guidelines to help developers in selecting the right architecture for the job. That being said, layered architectures have proven to be increasingly popular, due to their flexibility and ability to operate at multiple levels of abstraction simultaneously.

The book *AI and Mobile Robots* [8.79] has several chapters on architectures that have influenced this chapter. Most text books in robotics [8.19, 80, 81] have

sections on robot architectures. For many years in the mid 1990s, the AAAI Spring Symposia on Artificial Intelligence had sessions devoted to robot architec-

tures, although proceedings from those symposia are not widely available. More information on GRACE can be found in [8.82–84].

References

- 8.1 I. Jacobson, G. Booch, J. Rumbaugh: *The Unified Software Development Process* (Addison Wesley Longman, Reading 1998)
- 8.2 J.S. Albus: RCS: A reference model architecture for intelligent systems, Working Notes: AAAI 1995 Spring Symposium on Lessons Learned from Implemented Software Architectures for Physical Agents (1995)
- 8.3 R.A. Brooks: A robust layered control system for a mobile robot, *IEEE J. Robot. Autom.* **2**(1), 14–23 (1986)
- 8.4 R.J. Firby: An Investigation into Reactive Planning in Complex Domains, Proc. of the Fifth National Conference on Artificial Intelligence (1987)
- 8.5 R. Simmons: Structured control for autonomous robots, *IEEE Trans. Robot. Autom.* **10**(1), 34–43 (1994)
- 8.6 J.J. Borrelly, E. Coste-Maniere, B. Espiau, K. Kapelos, R. Pissard-Gibollet, D. Simon, N. Turro: The ORCAD architecture, *Int. J. Robot. Res.* **17**(4), 338–359 (1998)
- 8.7 B. Kuipers: The spatial semantic hierarchy, *Artif. Intell.* **119**, 191–233 (2000)
- 8.8 R. Orfali, D. Harkey: *Client/Server Programming with JAVA and CORBA* (Wiley, New York 1997)
- 8.9 R. Simmons, G. Whelan: Visualization Tools for Validating Software of Autonomous Spacecraft, Proc. of International Symposium on Artificial Intelligence, Robotics and Automation in Space (Tokyo 1997)
- 8.10 R. A. Brooks: The Behavior Language: User's Guide, Technical Report AIM-1227, MIT Artificial Intelligence Lab (1990)
- 8.11 R.J. Firby, M.G. Slack: Task execution: Interfacing to reactive skill networks, Working Notes: AAAI Spring Symposium on Lessons Learned from Implemented Architecture for Physical Agents (Stanford 1995)
- 8.12 E. Gat: ESL: A Language for Supporting Robust Plan Execution in Embedded Autonomous Agents, Proc. of the IEEE Aerospace Conference (1997)
- 8.13 V. Verma, T. Estlin, A. Jónsson, C. Pasareanu, R. Simmons, K. Tso: Plan Execution Interchange Language (PLEXIL) for Executable Plans and Command Sequences, Proc. 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (Munich 2005)
- 8.14 S.A. Schneider, V.W. Chen, G. Pardo-Castellote, H.H. Wang: ControlShell: A Software Architecture for Complex Electromechanical Systems, *Int. J. Robot. Res.* **17**(4), 360–380 (1998)
- 8.15 National Instruments: *LabVIEW* (National Instruments, Austin 2007), <http://www.ni.com/labview/>
- 8.16 N.J. Nilsson: A Mobile Automaton: An Application of AI Techniques, Proc. of the First International Joint Conference on Artificial Intelligence (Morgan Kaufmann Publishers, San Francisco 1969) pp. 509–520
- 8.17 N.J. Nilsson: *Principles of Artificial Intelligence* (Tioga, Palo Alto 1980)
- 8.18 P.E. Agre, D. Chapman: Pengi: An implementation of a theory of activity, Proc. of the Fifth National Conference on Artificial Intelligence (1987)
- 8.19 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)
- 8.20 J.H. Connell: SSS: A Hybrid Architecture Applied to Robot Navigation, Proc. IEEE International Conference on Robotics and Automation (1992) pp. 2719–2724
- 8.21 M. Mataric: Integration of Representation into Goal-Driven Behavior-Based Robots, Proc. IEEE International Conference on Robotics and Automation (1992)
- 8.22 I. Horswill: Polly: A Vision-Based Artificial Agent, Proc. of the National Conference on Artificial Intelligence (AAAI) (1993)
- 8.23 D.W. Payton: An Architecture for Reflexive Autonomous Vehicle Control, Proc. IEEE International Conference on Robotics and Automation (1986)
- 8.24 J.K. Rosenblatt: DAMN: A Distributed Architecture for Mobile Robot Navigation. Ph.D. Thesis (Carnegie Mellon Univ., Pittsburgh 1997)
- 8.25 R.C. Arkin: Motor schema-based mobile robot navigation, *Int. J. Robot. Res.* **8**(4), 92–112 (1989)
- 8.26 M. Arbib: Schema Theory. In: *Encyclopedia of Artificial Intelligence*, ed. by S. Shapiro (Wiley, New York 1992) pp. 1427–1443
- 8.27 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Proc. of the IEEE International Conference on Robotics and Automation (1985) pp. 500–505
- 8.28 R.C. Arkin: Integrating behavioral, perceptual, and world knowledge in reactive navigation, *Robot. Autonom. Syst.* **6**, 105–122 (1990)
- 8.29 R.C. Arkin, T. Balch: AuRA: Principles and practice in review, *J. Exp. Theor. Artif. Intell.* **9**(2/3), 175–188 (1997)
- 8.30 R.J. Firby: Adaptive Execution in Complex Dynamic Worlds. Ph.D. Thesis (Yale Univ., New Haven 1989)
- 8.31 R.J. Firby: Task Networks for Controlling Continuous Processes, Proc. of the Second International Conference on AI Planning Systems (1994)
- 8.32 R.P. Bonasso: Integrating Reaction Plans and layered competences through synchronous control, Proc. International Joint Conferences on Artificial Intelligence (1991)

- 8.33 S.J. Rosenschein, L.P. Kaelbling: The synthesis of digital machines with provable epistemic properties, Proc. of the Conference on Theoretical Aspects of Reasoning About Knowledge (1998)
- 8.34 L.P. Kaelbling: Goals as parallel program specifications, Proc. of the Sixth National Conference on Artificial Intelligence (1988)
- 8.35 L. P. Kaelbling: Compiling Operator Descriptions into Reactive Strategies Using Goal Regression, Technical Report, Teleos Research, TR90-10, (1990)
- 8.36 R.P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D.P. Miller, M.G. Slack: Experiences with an architecture for intelligent, reactive agents, J. Exp. Theor. Artif. Intell. **9**(2/3), 237–256 (1997)
- 8.37 E. Gat: Integrating Planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots, Proc. of the National Conference on Artificial Intelligence (AAAI) (1992)
- 8.38 G.N. Saridis: Architectures for Intelligent Controls. In: *Intelligent Control Systems: Theory and Applications*, ed. by Gupta, Sinhm (IEEE Press, Piscataway 1995)
- 8.39 R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingrand: An architecture for autonomy, Int. J. Robot. Res. **17**(4), 315–337 (1998)
- 8.40 M. Ghallab, H. Laruelle: Representation and control in IxTeT, a temporal planner, Proc. of AIPS-94 (1994)
- 8.41 P. Laborie, M. Ghallab: Planning with sharable resource constraints, Proc. of the International Joint Conference on Artificial Intelligence (1995)
- 8.42 M.P. Georgeff, F.F. Ingrand: Decision-Making in an Embedded Reasoning System, Proc. of International Joint Conference on Artificial Intelligence (1989) pp. 972–978
- 8.43 F. Ingrand, R. Chatila, R. Alami, F. Robert: PRS: A high level supervision and control language for autonomous mobile robots, Proc. of the IEEE International Conference On Robotics and Automation (1996)
- 8.44 N.P. Muscettola, P. Nayak, B. Pell, B.C. Williams: Remote agent: To boldly go where no AI system has gone before, Artif. Intell. **103**(1), 5–47 (1998)
- 8.45 B.C. Williams, P.P. Nayak: A Model-based Approach to Reactive Self-Configuring Systems, Proc. of AAAI (1996)
- 8.46 J.S. Albus: Outline for a theory of intelligence, IEEE Trans. Syst. Man Cybernet. **21**(3), 473–509 (1991)
- 8.47 B. Sellner, F.W. Heger, L.M. Hiatt, R. Simmons, S. Singh: Coordinated Multi-Agent Teams and Sliding Autonomy for Large-Scale Assembly, Proc IEEE **94**(7), 1425–1444 (2006), special issue on multi-agent systems
- 8.48 D. Goldberg, V. Ciciello, M.B. Dias, R. Simmons, S. Smith, A. Stentz: Market-Based Multi-Robot Planning in a Distributed Layered Architecture. In: *Multi-Robot Systems: From Swarms to Intelligent Automata*, Vol. II, ed. by A. Schultz, L. Parker, F.E. Schneider (Kluwer, Dordrecht 2003)
- 8.49 J.S. Albus, R. Lumia, H.G. McCain: NASA/NBS Standard Reference model for Telerobot Control System Architecture (NASREM), National Bureau of Standards, Tech Note #1235, NASA SS-GFSC-0027 (1986)
- 8.50 D.R. Blidberg, S.G. Chappell: Guidance and control architecture for the EAVE vehicle, IEEE J. Ocean Eng. **11**(4), 449–461 (1986)
- 8.51 R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras, H. Das: The CLARAty architecture for robotic autonomy, Proc. of the IEEE Aerospace Conference (Big Sky 2001)
- 8.52 I.A. Nesnas, R. Simmons, D. Gaines, C. Kunz, A. Diaz-Calderon, T. Estlin, R. Madison, J. Guineau, M. McHenry, I. Shu, D. Apfelbaum: CLARAty: Challenges and steps toward reusable robotic software, Int. J. Adv. Robot. Syst. **3**(1), 023–030 (2006)
- 8.53 T. Estlin, D. Gaines, C. Chouinard, F. Fisher, R. Castaño, M. Judd, R. Anderson, I. Nesnas: Enabling Autonomous Rover Science Through Dynamic Planning and Scheduling, Proc. of IEEE Aerospace Conference (Big Sky 2005)
- 8.54 R. Knight, G. Rabideau, S. Chien, B. Engelhardt, R. Sherwood: CASPER: Space Exploration through Continuous Planning, IEEE Intell. Syst. **16**(5), 70–75 (2001)
- 8.55 R. Simmons, D. Apfelbaum: A Task Description Language for Robot Control, Proc. of Conference on Intelligent Robotics and Systems (Vancouver 1998)
- 8.56 T.A. Estlin, D. Gaines, C. Chouinard, R. Castaño, B. Bornstein, M. Judd, I.A.D. Nesnas, R. Anderson: Increased Mars Rover Autonomy using AI Planning, Scheduling and Execution, Proc. of the International Conference On Robotics and Automation (2007) pp. 4911–4918
- 8.57 D. Musliner, E. Durfee, K. Shin: World modeling for dynamic construction of real-time control plans, Artif. Intell. **74**(1), 83–127 (1995)
- 8.58 D.J. Musliner, R.P. Goldman, M.J. Pelican: Using Model Checking to Guarantee Safety in Automatically-Synthesized Real-Time Controllers, Proc. of International Conference on Robotics and Automation (2000)
- 8.59 B. Espiau, K. Kapellos, M. Jourdan: Formal Verification in Robotics: Why and How?, Proc. International Symposium on Robotics Research (Hersching 1995)
- 8.60 G. Berry, G. Gonthier: The Esterel synchronous programming language: Design, semantics, implementation, Sci. Comput. Program. **19**(2), 87–152 (1992)
- 8.61 M. Jourdan, F. Maraninchi, A. Olivero: Verifying quantitative real-time properties of synchronous programs, Proc. 5th International Conference on Computer-aided Verification (Springer, Elounda 1993), LNCS 697
- 8.62 G. Pardo-Castellote, S.A. Schneider: The Network Data Delivery Service: Real-Time Data Connectivity for Distributed Control Applications, Proc. of Inter-

- national Conference on Robotics and Automation (1994) pp. 2870–2876
- 8.63 JAUS Reference Architecture Specification, Volume II, Part 1 Version 3.2 (available at <http://www.jauswg.org/baseline/refarch.html>)
- 8.64 JAUS Tutorial Powerpoint slides (available at: <http://www.jauswg.org/>)
- 8.65 JAUS Domain Model Volume I, Version 3.2 (available at http://www.jauswg.org/baseline/current_baseline.shtml)
- 8.66 E. Gat: ALFA: A Language for Programming Reactive Robotic Control Systems, Proc. IEEE International Conference on Robotics and Automation (1991) pp. 116–1121
- 8.67 R.A. Brooks: Elephants don't play chess, J. Robot. Autonom. Syst. **6**, 3–15 (1990)
- 8.68 L.P. Kaelbling: Rex– A symbolic language for the design and parallel implementation of embedded systems, Proc. of the 6th AAAI Computers in Aerospace Conference (Wakefield 1987)
- 8.69 E. Gat: Non-Linear Sequencing and Cognizant Failure, Proc. AIP Conference (1999)
- 8.70 E. Gat: On the role of stored internal state in the control of autonomous mobile robots, AI Mag. **14**(1), 64–73 (1993)
- 8.71 J.L. Peterson: *Petri Net Theory and the Modeling of Systems* (Prentice Hall, Upper Saddle River 1981)
- 8.72 K. Currie, A. Tate: 0-Plan: The open planning architecture, Artif. Intell. **52**(1), 49–86 (1991)
- 8.73 D.S. Nau, Y. Cao, A. Lotem, H. Muñoz-Avila: SHOP: Simple hierarchical ordered planner, Proc. of the International Joint Conference on Artificial Intelligence (1999) pp. 968–973
- 8.74 S. Chien, R. Knight, A. Stechert, R. Sherwood, G. Rabideau: Using iterative repair to improve the responsiveness of planning and scheduling, Proc. of the International Conference on AI Planning and Scheduling (2000) pp. 300–307
- 8.75 N. Muscettola: HSTS: Integrating planning and scheduling. In: *Intelligent Scheduling*, ed. by M. Fox, M. Zweben (Morgan Kaufmann, San Francisco 1994)
- 8.76 R. Simmons, J. Fernandez, R. Goodwin, S. Koenig, J. O'Sullivan: Lessons Learned From Xavier, IEEE Robot. Autom. Mag. **7**(2), 33–39 (2000)
- 8.77 R. Simmons: *Inter Process Communication* (Carnegie Mellon Univ., Pittsburgh 2007), www.cs.cmu.edu/IPC
- 8.78 S.W. Ambler: *UML 2 Sequence Diagrams* (Ambisoft, Toronto 2007), www.agilemodeling.com/artifacts/sequenceDiagram.htm
- 8.79 D. Kortenkamp, R.P. Bonasso, R. Murphy: *Artificial Intelligence and Mobile Robots* (AAAI Press/The MIT Press, Cambridge 1998)
- 8.80 R. Murphy: *Introduction to AI Robotics* (MIT Press, Cambridge 2000)
- 8.81 R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2004)
- 8.82 R. Simmons, D. Goldberg, A. Goode, M. Montemerlo, N. Roy, B. Sellner, C. Urmson, A. Schultz, M. Abramson, W. Adams, A. Atrash, M. Bugajska, M. Coblenz, M. MacMahon, D. Perzanowski, I. Horswill, R. Zubeck, D. Kortenkamp, B. Wolfe, T. Milam, B. Maxwell: GRACE: An autonomous robot for the AAAI Robot Challenge, AAAI Mag. **24**(2), 51–72 (2003)
- 8.83 R. Gockley, R. Simmons, J. Wang, D. Busquets, C. DiSalvo, K. Caffrey, S. Rosenthal, J. Mink, S. Thomas, W. Adams, T. Lauducci, M. Bugajska, D. Perzanowski, A. Schultz: Grace and George: Social Robots at AAAI, AAAI 2004 Mobile Robot Competition Workshop (AAAI Press, 2004), Technical Report WS-04-11, pp. 15–20
- 8.84 M.P. Michalowski, S. Sabanovic, C. DiSalvo, D. Busquets, L.M. Hiatt, N.A. Melchior, R. Simmons: Socially Distributed Perception: GRACE plays social tag at AAAI 2005, Auton. Robot. **22**(4), 385–397 (2007)

AI Reasoning

9. AI Reasoning Methods for Robotics

Joachim Hertzberg, Raja Chatila

Artificial intelligence (AI) reasoning technology involving, e.g., inference, planning, and learning, has a track record with a healthy number of successful applications. So, can it be used as a toolbox of methods for autonomous mobile robots? Not necessarily, as reasoning on a mobile robot about its dynamic, partially known environment may differ substantially from that in knowledge-based pure software systems, where most of the named successes have been registered.

This Chapter sketches the main robotics-relevant topics of symbol-based AI reasoning. Basic methods of knowledge representation and inference are described in general, covering both logic- and probability-based approaches. Then, some robotics-related particularities are addressed specially: issues in logic-based high-level robot control, fuzzy logics, and reasoning under time constraints. Two generic applications of reasoning are then described in some detail: action planning and learning.

General reasoning is currently not a standard feature onboard autonomous mobile robots. Beyond sketching the state of the art in robotics-related AI reasoning, this Chapter points to the involved research problems that remain to be solved towards that end.

The Chapter first reviews knowledge representation and deduction in general (Sect. 9.1), and

9.1 Knowledge Representation and Inference	208
9.1.1 Logic	208
9.1.2 Probability Theory	210
9.2 KR Issues for Robots	212
9.2.1 Logics for High-Level Robot Control	212
9.2.2 Fuzzy Logic Approaches	213
9.2.3 Reasoning under Time Constraints	214
9.3 Action Planning	214
9.3.1 Planning Domain Descriptions	215
9.3.2 Partial-Order Plan Generation	216
9.3.3 Planning Under Uncertainty	218
9.3.4 Robot Planning	219
9.4 Robot Learning	219
9.4.1 Inductive Logic Learning	220
9.4.2 Statistical Learning and Neural Networks	220
9.4.3 Reinforcement Learning	220
9.5 Conclusions and Further Reading	221
References	222

then goes into some detail regarding reasoning issues that are considered particularly relevant for applications in robots (Sect. 9.2). Having presented reasoning *methods*, we then enter the field of generic reasoning *applications*, namely, action planning (Sect. 9.3) and machine learning (Sect. 9.4). Section 9.5 concludes.

Historical debates about the necessity and wisdom of employing symbolic reasoning in autonomous mobile robots notwithstanding, there now seems to be agreement that some part or layer in the control system of such robots should or could include reasoning. Hybrid control architectures (Chap. 8) would be typical software structures for solving the difficult problem of amalga-

inating their contributions with those of other parts of the controller, while guaranteeing sufficiently short control cycle times, so that the robot can act safely in a dynamic environment.

Symbolic reasoning is understood here in the classical sense of artificial intelligence (AI), i. e., deliberation based on symbols as in first-order predicate logic

(FOPL) or Bayesian probability theory, but typically with restrictions and/or extensions thereof, which have

to be traded off against each other to achieve a suitable combination of expressivity and inference speed.

9.1 Knowledge Representation and Inference

Reasoning requires that the reasoner – in our case, a robot – has an explicit representation of parts or aspects of its environment to reason about. Two questions come up immediately: what are suitable formats for such an explicit representation, and where does the represented knowledge come from?

The second question refers to the problem of generating and maintaining in real time a symbolic description of the robot's environment, or at least of some part of it, based on a previous symbolic description and on recent environment information obtained from sensors and communication with other agents. In full generality, this problem is currently unsolved, involving AI fundamentals such as *symbol grounding* [9.1] and *object anchoring* [9.2]. So practical symbolic reasoning in a robot is restricted to that part of its knowledge that can be kept sufficiently recent. This includes, obviously, static knowledge about the environment, such as the topological elements and their relations in a building; transient knowledge available in symbolic forms, such as in facility management databases; and, most challenging, symbolic data that has to be distilled from sensor data. Object recognition from camera data (Chap. 23) is a relevant method here.

This section deals with answers to the first question, i.e., formalisms suitable for representing knowledge. Suitability has two aspects here that are to be treated as two sides of one coin. One is *epistemological adequacy*: does the formalism allow the targeted aspects of the environment to be expressed compactly and precisely? The other is *computational adequacy*: does the formalism allow typical inferences to be drawn effectively and efficiently? There is a tradeoff between these two adequacies, given that very rich, expressive, and hence epistemologically appealing formalisms are typically accompanied by intractable or even undecidable inference problems, and vice versa. Knowledge representation (KR), then,

is the field of AI that focuses on the design of formalisms that are both epistemologically and computationally adequate for expressing knowledge about a particular domain. [9.3, p. xiii]

In this section, we focus on two families of formalisms, namely, logic and probability theory, and respective inference procedures. For introductions to the field of KR and for sources for recent results, we refer to the general AI and KR literature as given at the end of this chapter. In addition, there is the *International Conference on Knowledge Representation (KR)*, held biannually every *even* year.

9.1.1 Logic

First-order predicate logic (FOPL) is the archetype of KR formalisms in AI. Having said this, let us add immediately that it does not qualify for such a formalism in the aforementioned sense of being both epistemologically and computationally adequate for typical application domains – in many cases, it is neither of the two. However, it is the background for conceptually and mathematically understanding formalisms for representing and reasoning with definite knowledge.

We assume familiarity with the basic notions of FOPL, and rather than giving another sketch here, we refer to the substantial body of textbooks and introductions. Every typical AI textbook introduces FOPL, and [9.4, 5] are no exceptions. [9.6] is a principled introduction to logics; [9.7] a practical one, and [9.8] a concise mathematical one.

Representing knowledge in some variant of logic facilitates drawing inferences from that knowledge using provably sound and complete calculi. Automated deduction is a subfield of logic and AI that offers a healthy number of very powerful implemented deduction systems that can readily be used. (See [9.4, Chap. 9] for an introduction; [9.9] is a recent, comprehensive source book.)

Relying on logical inference, a robot could deduce a large number of facts that might otherwise be hard or impossible to obtain. For example, assume that a robot perceives by acquiring and interpreting sensor data, that the door D_{509} in an office building is currently closed: $Closed(D_{509})$ expressed in some ad hoc FOPL language, assuming intuitive interpretations of symbols. Let us further assume that the robot's base of static knowledge

about the building contains the sentences

$$\begin{aligned}
 & \text{Connects}(D_{509}, C_5, R_{509}) , \\
 & \text{Connects}(D_{508}, C_5, R_{508}) , \\
 & \text{Connects}(D_{508a}, R_{508}, R_{509}) , \\
 & \forall d, l_1, l_2 [\text{Connects}(d, l_1, l_2) \\
 & \qquad \qquad \qquad \leftrightarrow \text{Connects}(d, l_2, l_1)] , \\
 & \forall d. [\text{Closed}(d) \leftrightarrow \neg \text{Open}(d)] , \\
 & \forall l. [\text{At}(l) \rightarrow \text{Accessible}(l)] , \\
 & \forall l_1, l_2. [\text{Accessible}(l_1) \\
 & \qquad \rightarrow (\exists d. [\text{Connects}(d, l_1, l_2) \wedge \text{Open}(d)] \\
 & \qquad \qquad \rightarrow \text{Accessible}(l_2))]
 \end{aligned}$$

Constants D_i and variable d denote doors; R_i denote rooms; C_i corridors; variables l, l_i denote locations, i.e., rooms and corridors. Assume that the robot's localization tells it its current room or corridor location as $\text{At}(\cdot)$.

Then, assuming the robot knows itself to be $\text{At}(C_5)$, observing $\text{Closed}(D_{509})$ entails $\neg \text{Open}(D_{509})$, and, more interesting, $\text{Accessible}(R_{509})$ only if $\text{Open}(D_{508}) \wedge \text{Open}(D_{508a})$ is true. So, performing, for example, a delivery task to room R_{509} , the robot may replan its route through R_{508} , unless at least one of D_{508} and D_{508a} is known to be closed. If the status of one or both of them is unknown (neither $\text{Open}(\cdot)$ nor $\text{Closed}(\cdot)$ is entailed by the current knowledge base), then accessibility of R_{509} can be neither proven nor disproven. That would leave open the option of planning the route through D_{508} and D_{508a} , gathering their required statuses on site.

So, as this little example shows, FOPL is a powerful representation and reasoning tool. Moreover, its theory is very well understood. In particular, it is well known that consequence in FOPL is in general undecidable, which means that a sound and complete deduction algorithm cannot even guarantee termination for some particular inference attempt, let alone speedy results.

However, that does not mean logic as a representation and inference tool needs to be completely disposed of. Many applications do not require the full expressivity of FOPL. Moreover, there are many interesting language subsets of FOPL that are decidable and can be handled by algorithms that are efficient in most practical cases. So some effort by the KR community has gone into identifying FOPL subsets and fitting inference procedures that qualify for being both epistemologically and computationally adequate for a broad number of applications. We will consider two of these in more detail: propositional logic and description logics.

Propositional Theories

In quite a number of practical cases, FOPL theories (sets of formulas) in fact represent finite domains. Logically speaking, they have finite Herbrand universes or can at least be recast in a form such that they have. In this case, it may still be handy to express the domain theory in FOPL syntax, but all that goes beyond a purely propositional theory is just for notational convenience. For example, an axiom stating that a robot can only be in one location (room or corridor) at a time

$$\forall l_1, l_2. [\text{At}(l_1) \rightarrow (\neg \text{At}(l_2) \vee l_1 = l_2)]$$

can be *flattened* for a finite building into the, clumsy but equivalent, form handling all locations explicitly, for example in a six-storey building

$$\begin{aligned}
 \text{At}(R_{001}) & \rightarrow [\neg \text{At}(R_{002}) \wedge \neg \text{At}(R_{003}) \wedge \dots \\
 & \qquad \neg \text{At}(R_{514}) \wedge \neg \text{At}(C_{0a}) \wedge \\
 & \qquad \neg \text{At}(C_{0b}) \wedge \dots \wedge \neg \text{At}(C_5)] , \\
 \text{At}(R_{002}) & \rightarrow [\neg \text{At}(R_{001}) \wedge \neg \text{At}(R_{003}) \wedge \dots \\
 & \qquad \neg \text{At}(R_{514}) \wedge \neg \text{At}(C_{0a}) \wedge \\
 & \qquad \neg \text{At}(C_{0b}) \wedge \dots \wedge \neg \text{At}(C_5)] , \\
 & \dots
 \end{aligned}$$

where every ground (variable-free) instance of a predicate, such as the At instances above, are to be considered as propositional variables, regarding textual identity.

The good news here is that the corresponding *flat* theory to a FOPL theory over a finite Herbrand universe is propositional; hence, it is decidable. Moreover, under some practical conditions – e.g., if the variables in the FOPL theory are sorted (that is, the information is available that the variables l_1, l_2 , e.g., range over rooms and corridors) – it can even be generated mechanically from a more compact FOPL syntax.

The potentially bad news is that the now propositional theory may of course consist of a huge number of propositional sentences: In general, as all combinations of variable substitutions in all FOPL sentences need to be generated, the growth is multi-exponential in terms of the domain sizes of the FOPL variables.

However, the technology for propositional satisfiability checking or model checking is making considerable progress, allowing propositional theories with thousands or even tens of thousands of variables to be handled in the order of seconds of computation time on regular hardware. These methods are particularly efficient if many models exist for a satisfiable propositional formula or if the truth or falsity of many ground facts is

known a priori (such as $At(C_5)$ for a robot by independent localization). Both are often the case in practical KR.

[9.4, Chap. 7] gives an introduction to model checking. One family of methods is based on the classical Davis–Putnam (DPLL) algorithm [9.10]. It attempts to construct systematically a propositional model of a given theory, efficiently propagating interpretation constraints for variables. Another family of algorithms applies local search techniques, attempting to generate interpretations using random (Monte Carlo) variable assignments. [9.11] is a recent collection of papers summarizing the state of the art, including a paper about the results of the SAT2002 competition that discusses in detail the performance of recent satisfiability checkers on synthetic (i.e., typically unnaturally hard) problems.

Description Logics

Description logics (DL) have emerged as a rigorous formalization of somewhat intuitive KR forms of the AI of the 1970s, namely, semantic networks and frame systems. Logically speaking, DLs, of which there is a considerable variety, form a certain family of decidable subsets of FOPL.

There are two parts of representing knowledge about a particular domain using a DL language. First, the *upper ontology* of the domain is formulated. It introduces the general domain concepts as well as relations between these concepts. A particularly interesting type of relations occurring in all ontologies is the superclass–subclass relation. Given that DLs – as part of the means for enforcing decidability – strictly disallow cyclic relations between concepts to be specified, the superclass relation imposes a hierarchical taxonomy on concepts, which serves to define property inheritance, much like in object-oriented programming. This first part of a DL-based domain representation is, for historical reasons, often called the *TBox* (or *terminological knowledge*).

A *concept* in the DL language corresponds to a unary predicate. To give an example, an ontology of robot navigation domains might include concepts like *Door*, *Location*, and so on. The concept hierarchy is built by defining concept equality = or a subconcept property, e.g., $Room \sqsubseteq Location$, and $Corridor \sqsubseteq Location$. Concepts can be combined using concept conjunction, disjunction and, negation (\sqcap , \sqcup , \neg , respectively), allowing, e.g., concept definitions such as $Location = Room \sqcup Corridor$, $Door = Closed \sqcup Open$, and $Open = \neg Closed$.

Roles in a DL language correspond to binary predicates, such as *leadsTo* for a door and a location. Inversity, intersection, and union of roles are defined as expected, where $leadsTo = leadsFrom^{-1}$ is an example for defining inverse roles. Roles can be composed, such as in defining $adjacent = leadsFrom \circ leadsTo$ (location l is adjacent to m if and only if some door leads from l to m). Finally, concepts and roles can be combined for defining new concepts and roles. In particular, it is possible to quantify over *role-fillers*, i.e., the individual objects (see below) that can be consistently substituted for role arguments, for example, one could define $BlockedLoc = Location \sqcap \neg \exists leadsFrom.Open$ (assuming the intuitive binding rules of the operators). Different variants of DLs differ in what other operators they make available. See [9.3] for details.

As the second part of domain representation using DLs, individual objects have to be introduced into the language of concepts and roles. This part of the domain representation is called *ABox* or *assertional knowledge*, for example, $Room(R_{509})$, $leadsTo(D_{509}, R_{509})$, and $Closed(D_{509})$ could be asserted.

DLs have a number of reasoning services to offer, which are based on logical inference in a given TBox and ABox. They include consistency of the concept definition, subsumption and disjointness of concepts, consistency of the ABox with respect to the TBox, concept and role instances, all of which are decidable in a DL. Given the TBox and ABox rudiments above, it would, e.g., be concluded that everything is consistent and that $BlockedLoc(R_{509})$ (note that only door D_{509} is known here to lead to R_{509}). These DL inferences are theoretically intractable, but run very efficiently in most practical cases.

Reference [9.3] provides a comprehensive overview of the state of the art in DL. In 2004, the WWW consortium (W3C) defined the web ontology language (OWL) [9.12] as a technical basis for the semantic web. Part of the language is OWL-DL, a classical DL in the sense just described. OWL-DL ontologies are starting to become publicly available over the web; see [9.13] for a tutorial introduction.

9.1.2 Probability Theory

KR formalisms based on logics are worth considering whenever factual knowledge is to be represented, from which deductions are to be made. Part of the knowledge that a robot might use about its environment does, however, not really have this character. Choosing among KR

formalisms is – just like choosing among programming languages – to some degree a matter of taste, experience, familiarity, or system integration considerations. In KR terms the epistemological adequacy of a formalism for some domain is not an issue that can be judged with precision.

However, logic, at least its classical variants, has a number of weak spots in terms of KR requirements for some application types. Handling *uncertainty* is one, or rather, a whole family of them, as uncertainty is in itself an overloaded concept. *Lack of knowledge* is one of its aspects. Logics can handle this insofar as truth or falsity of some facts may remain undetermined. However, if too much is undetermined in a knowledge base, logics will no longer be able to make interesting deductions, as *everything is possible* logically. However, intuitively the different possibilities may differ considerably in likelihood.

The field of KR has adopted Bayesian probability as a means of representing and reasoning with this type of uncertainty, using correlations between facts rather than strict implications. Note that this approach amalgamates different sources of lack of precise and complete knowledge. Some piece of knowledge may be unknown because it is in principle unknowable, or because it was judged too costly to build a precise theory or determine all information relevant for making a sound deduction. In either case, working with probabilities rather than binary truth values can serve as an approximation. Note that the conception of truth is still the same here as in classical logics: *objectively*, a fact is supposed to be either true or false; probability just models the subjective degree of belief that the fact is true. Note that this is a difference from fuzzy logics, which will be addressed briefly below (Sect. 9.2.2).

In analogy to Sect. 9.1.1, we here assume familiarity with the basic notions of probability theory. Reference [9.4, Chap. 13] gives an excellent basic introduction; for a more comprehensive treatment, there are a large variety of introductory textbooks around, of which [9.14] is a recent example.

Inference in Bayesian probability theory basically means to infer the probability of some event of interest, given other prior and dependent relevant probabilities. Practically, an important type of probabilistic inference is *diagnostic reasoning*: reasoning from observed effects back to hidden causes, given causal rules that specify conditional probabilities from causes to effects. So the problem is, for a potential cause C and an observed effect E : given prior probabilities $P(C)$ and $P(E)$ and the conditional probability $P(E|C)$, determine the posterior

$P(C|E)$. The solution is of course given by the Bayes rule as

$$P(C|E) = \frac{P(E|C)P(C)}{P(E)}.$$

However, as in logical reasoning, this theoretically appealing principle turns out to be impractical if applied naively. Consider that not just one effect E may be observed, but n of them; moreover, not all of these are conditionally independent. A generalized form of the Bayes rule to calculate the correct posterior is straightforward, but who can specify all the conditional probabilities involved – in the worst case $O(2^n)$ of them, where n may easily range in the hundreds in practical cases?

Until the late 1980s, this problem has more or less been circumvented. One way is to treat the E_i in bad faith as independent, simply use the n individual conditional probabilities $P(E_i|C)$ and use them straightforwardly to approximate the full joint probability distributions. Reference [9.4, Chap. 14.7] reviews this and other alternative approaches.

The solution used ever since it has appeared [9.15] is *Bayes networks* (BNs). The idea is to represent the random variables in a directed acyclic graph, where a node is directly preceded by a set of parent nodes iff it is directly conditionally dependent on the corresponding parent variables. So the huge full joint probability distribution is broken down into many, typically very small, *local joint probability distributions* without loss of information, the trick being to use the typically many known conditional independencies among variables to reduce the representational and inferential effort drastically.

Figure 9.1 shows a simple BN expressing that D can be caused by B or C , with probabilities given by a conditional probability table that specifies locally the joint probability distribution. In addition, the structure says

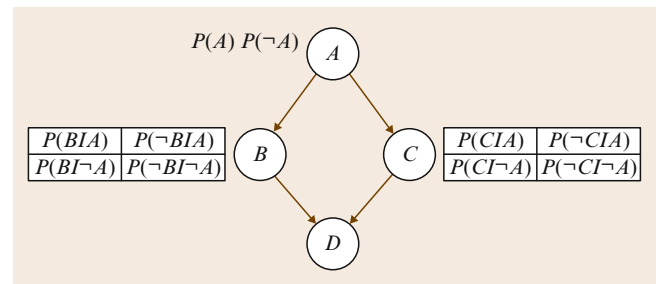


Fig. 9.1 Structure of a simple Bayesian network associated with conditional probability tables. (The ones for D , dependent on B and on C , are omitted.)

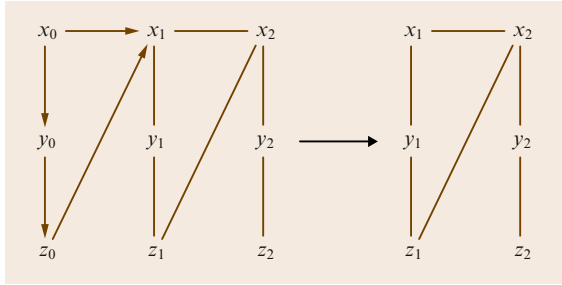


Fig. 9.2 Example of a dynamic Bayesian network at two successive time steps. The structure is invariant over time

that D is independent from A knowing B and C (a node is independent from its ancestors knowing its parents), and that B is independent from C knowing A , i.e., that $P(C|A, B) = P(C|A)$ and $P(B|A, C) = P(B|A)$. The

9.2 KR Issues for Robots

The reasoning methods and generic applications in the rest of this chapter all make use, to some degree, of variants of the basic formalisms of logic or Bayesian probability theory. Both formalisms, in particular if variants are included, are so general and powerful that they can contribute to any KR application.

However, reasoning in a robot control system differs from reasoning in other knowledge-based systems in a number of aspects. A knowledge-based robot is an embedded system, working in closed-loop control, typically in a dynamic environment with quite limited resources; it has to interpret itself its sensor input, and should execute at least some of the actions autonomously that it has judged appropriate. Accordingly, knowledge-based robot control has special needs that are not present in, say, an offline expert system, which gets its input by prompting an experienced human in its domain of expertise and writes its output into some document for more experienced humans to deliberate about and possibly act accordingly if they feel it right. Naturally, this difference poses particular demands of robot control on the side of KR that may not be of interest for many other KR customers and that lie outside the mainstream of the field. In this section, we very briefly address three topics.

Before doing so, a word about the role of representation in designing *intelligent* or *smart* robots. It has been convincingly demonstrated that some forms of *intelligence without representation* [9.16] can be achieved in robots; Chap. 38 provides the background to this.

network can be exploited in both directions. Bottom up, it enables one to explain an observed phenomenon using the known conditional probabilities (diagnostic). For example, if D is observed, its more probable cause (B or C) can be inferred. Top down, it enables one to propagate evidence to compute the probability of a phenomenon, e.g., to compute the probability of D given A (causation).

For systems evolving over time, it is important to use an invariant structure of the Bayes network in order to update the values of the variables while using the same reasoning process. Such networks are called dynamic Bayesian networks (DBNs). Figure 9.2 shows such a network at time T , then at time $T + 1$.

In Sect. 9.3.3 we shall see specific probabilistic techniques for planning under uncertainty such as Markov decision processes.

Clearly, the most sophisticated representation and inference system is of no help for a robot, if the reasoning in terms of symbols cannot be grounded in real sensor data and actuator control. On the other hand, representation and inference have much to offer for improving the performance, the software design, and the user interface of robots, for example in field, service, and intervention applications (Part F). Hybrid robot control systems (Chap. 8) are currently the means for amalgamating instances of KR technology with reactive control elements in a robot.

9.2.1 Logics for High-Level Robot Control

Robot domains are dynamic by nature, including at least one physical agent, namely, the robot. Capturing this in a logic-based formalism poses some conceptual and technical problems, which are introduced in [9.4, Chap. 10.3].

Very briefly, one of these comes from the requirement of expressing concisely and reasoning efficiently with logical models of action. Logically speaking, an individual action changes the truth value of a limited number of facts, leaving everything else as it was. For example, modeling on some abstract level the action of going from one location to another as atomic, it changes the robot's current location before and after; depending on the modelization, it may also change the battery state and the mileage count; but it does not change the lay-

out of the building or the name of the president. The problem of formalizing actions concisely in a logical language so that they allow facts that may or may not have changed after applying a sequence of actions to be inferred efficiently, has been termed the *frame problem*. It has received much attention in the literature; there are now a number of practical solutions to it.

Another problem concerns knowledge-base update, made necessary by independently changing facts. Consider, for example, a robot sitting – as it is told by its self-localization – in front of some door *D*, which is believed to be open, but the robot perceives a closed door. Logically, this is a contradiction. Now there are in theory several ways to make the knowledge and the perception consistent. One is to assume that *D* got closed since learning that it was open – probably the most intuitive explanation. Logically just as good would be, e.g., that the perception is faulty, or that the robot has been teleported in front of a door that is known to be closed. Among these *explanations*, some are more intuitively plausible than others; logically, some would require less formulas of the knowledge-base to be withdrawn and should therefore be preferred. Ideally, after replacing an old information by a new one, one would have to make sure that the consequences of a retracted formula are no longer believed.

Theoretically, these problems are arbitrarily difficult. Practically, they can be sufficiently restricted to allow solutions within a neat logical framework. Typical solutions would introduce some notion of situation or holding period for formulas, following the classical example of the situation calculus [9.17]. This allows change to be tracked. Typical solutions would also give up completeness of inference, resorting to a PROLOG-like inference engine. Three examples for such solutions with reported applications in robot control are GOLOG [9.18], event calculus [9.19], and FLUX [9.20].

9.2.2 Fuzzy Logic Approaches

For historical reasons, a number of methods and techniques dealing with classical AI questions have been living outside the *inner circle* of AI, sometimes under the different field names of *soft computing* or *computational intelligence*. This relates in particular to *fuzzy logic*, which is clearly a symbolic reasoning method. Moreover, it shares its fundamentals with fuzzy control, which gets used, in one way or another, in quite a number of robot controllers, so it is certainly a formalism to consider for KR on a robot.

To start with, the ontological basis of fuzzy logic is different to that of FOPL and probability theory. The latter assume that predicates are *objectively* either true or false; the truth may not be known, but it is in fact there. Fuzzy logic assumes that truth *objectively* comes in degrees. Many natural language categories have a fuzzy flavor, e.g., the statement that some person is *big* could be interpreted as objectively true or false in some extreme cases – but what should one *objectively* say about a male of 185 cm and 80 kg?

Fuzzy logic is conceived as a generalization of propositional logic (often called *crisp* logic in the field). Consequently, fuzzy logic uses the regular junctors \wedge , \vee , \neg etc., but generalizes their truth functional definitions from the set $\{0, 1\}$ to the interval $[0, 1]$. There are several plausible ways of doing this; a frequent one is to define

$$\begin{aligned}\text{value}(\neg P) &= 1 - \text{value}(P) \\ \text{value}(P \vee Q) &= \max\{\text{value}(P), \text{value}(Q)\}.\end{aligned}$$

The other junctors can be defined in the same spirit, where a number of differing definitions are in use.

Fuzzy deduction then typically has a different purpose, and hence a different machinery than propositional deduction – satisfiability of some fuzzy logic theory is normally not an interesting question as it too would come in degrees, so most fuzzy theories would normally be at least *a bit satisfiable*. Fuzzy logic knowledge bases are normally applied in a forward chaining manner over sets of fuzzy inference rules, inferring fuzzy values of fuzzy variables from other known fuzzy values. This is how fuzzy knowledge bases are successfully used in robot controllers, e.g., in the Saphira control architecture [9.21]. For example, it would make sense to determine the steering angle of a mobile robot using rules like

```
if    (freeRange right is_narrow)  $\wedge$ 
      (freeRange front is_medium)  $\wedge$ 
      (freeRange left is_wide)  $\wedge$ 
then  set angle medium_left
```

where the italicized components (*right*, *is_narrow* etc.) are fuzzy variables. The fuzzy truth values of the antecedent variables at a particular instance in time would be determined, e.g., from the readings of a distance sensor via definitions of the variables mapping, e.g., *is_medium* to a normal distribution with mean of 2 m; the set angle *medium_left* might be mapped to a normal distribution with mean 30° to the left. The actual

set angle to apply would finally be calculated by *defuzzification* of the combined fuzzy set values of the best applicable rule or rules; a plausible scalar value would be the center of gravity of the resulting fuzzy value.

Weltanschauung debates between soft computing and *core AI* notwithstanding, fuzzy reasoning components and other *AI* components can live together well in a robot controller, each contributing their respective strengths. The Saphira architecture is an example.

9.2.3 Reasoning under Time Constraints

Unlike many usual *AI* systems, robots have to take decisions in real time based on sensory data, in order to cope with the dynamics of their environments. Hence their decisions should be produced in limited time. If these decisions require some lookahead and anticipation, i.e., if planning is required, a problem arises: real-time decision-making is indeed incompatible with the unbounded nature of planning, which in its general form is an intractable problem, and being based on *FOPL* inference mechanisms, is in addition only semi-decidable.

In order to take real-time constraints into account, approaches such as reactive planning have been proposed [9.22], which actually abandons to plan too much ahead (only one step at a time). However, the solution

to time constraints usually comes from the *architectural design* of the robot system (Chap. 8), which integrates a decision-making level and a reactive level that would react within a bounded time. The decision-making level often includes a planner and a supervisory control system that ensures task execution supervision using predefined procedures or scripts, with explicit triggering conditions. This supervisory component has to be both goal and event driven: goal driven to accomplish the task and event driven to be reactive to environment changes. An example of such a system is the procedural reasoning system (*PRS* [9.23, 24]), which includes a set of procedures, a database updated by sensing and by the evolution of system state, and an interpreter that decides to launch the applicable procedures according to the contents of the database.

Anytime algorithms [9.25, 26] were developed as a possible response to time constraints. The basic idea is that the longer the algorithm can operate, the closer to optimal its solution will be. However, if time is shortened, the algorithm will still produce an answer, which may be far from optimal, but will enable the system to act instead of being blocked awaiting a decision. Anytime algorithms evaluate the quality of their results and a performance profile enables one to predict how this quality can be improved over time.

9.3 Action Planning

For the rest of this Chapter, we turn from reasoning *methods* to two generic reasoning *applications*, namely action planning and learning.

Planning, in the traditional *AI* sense, means deliberating about a course of action for an agent to take for achieving a given set of goals. *Scheduling* means allocating time and potentially other resources to a set of actions such that given deadlines are met and resource constraints respected. The two activities are conceptually different and have historically often been treated in a cascaded way: plan first, then schedule. Recently, with the advent of more efficient temporal reasoning methods and more efficient planning algorithms, there is a tendency to deal with them in an integrated way. The rationale is that generating an optimal plan according to some metric first, and then designing an optimal schedule for that plan, may not lead to a time- and resource-optimal schedule. It does still also make sense, however, to consider the two in isolation as their combi-

nation is not always required. In this overview, we skip the integration of scheduling. Reference [9.27, Part IV] provides a comprehensive overview.

Robot control has been among the first intended applications of planning: the STRIPS system [9.28, 29] was used to generate plans, i.e., sequences of abstract high-level actions to execute, for the robot SHAKEY [9.30]. However, it has turned out to be nontrivial to design a robot control architecture in which plans containing abstract actions are broken down into physical operation that is flexible enough to follow the plan but cope with contingency, serendipity, and failure as one would intuitively expect from an *intelligent* usage of plans.

This overview starts with how to model planning domains for planners. In Sect. 9.3.2, we sketch planning algorithms that work efficiently under a number of simplifying assumptions. Section 9.3.3 relaxes these assumptions, dealing with uncertainty such as lack of

observability. We close with a brief discussion of robot planning Sect. 9.3.4.

For comprehensive coverage of the topic, we refer to [9.27]. Introductions to the topic can be found in AI textbooks, such as [9.4, Chap. 11,12,16,17]. Reference [9.31] concisely reviews the more recent developments in planning. The *International Conference on Automated Planning and Scheduling* (ICAPS) is currently the main international conference of the field, held annually. The PLANET online research database [9.32, Service/Repositories] contains a set of continually updated links to state-of-the-art planners, schedulers, and related tools. Reference [9.33] contains an application-oriented perspective of the current state and perspectives of the field as of the end of 2003.

9.3.1 Planning Domain Descriptions

Deliberating about the course of action of an agent requires representing its possible actions, including their effects, keeping track of changes in the environment over time, and dealing rationally with any remaining uncertainty or lack of information. In this sense, planning combines the reasoning methods described previously in this Chapter.

The extreme form of deliberation would be to simulate in advance the environment, including the agent's possible actions, and then do what seems best. In practice, that is impossible for at least two reasons. First, typically not all information that would be needed for a truthful simulation is available. Planning is meant for real environments in which many parameters are unknown or unknowable and which are not under the agent's control. Second, even if everything for a complete simulation were known, then it would very likely be so computationally intensive that the real world would run ahead of the simulation all the time. In consequence, planning domain representations deliberately abstract away many, or most, details, to make effective planning possible. As in all forms of KR, precision and completeness of domain models need to be traded off against reasoning speed.

Exactly how many or how few details, and which, are represented, is an engineering decision to take when a planning system is built. Accordingly, a number of simplifying assumptions on the domain may or may not be made, including

- *finiteness* (the domain has only finitely many objects, actions, and states)

- *information completeness* (the planner has all relevant information at planning time)
- *determinism* (actions have deterministic effects)
- *instantaneousness* (actions have no relevant duration)
- *idleness* (the environment does not change during planning)

As usual, restrictive assumptions lead to efficient algorithms, but may cause problems when a simplified solution makes contact with the real world – in this case at execution time. Note that using a simplifying domain model does *not* mean assuming that the world objectively *is* that way. It may, e.g., make sense to plan only for standard cases, assuming maximal simplicity, and handle any occurring exceptions at execution time if and when they occur. In robot planning, this philosophy goes back as far as SHAKEY [9.29].

Currently, two main families of domain representations are used. The PDDL-type description is described next. In Sect. 9.3.3, a representation for nondeterministic domains will be sketched. In addition, there are various forms of deductive planning orthogonal to these two that use variants of *pure* logical representations. See [9.27, Chap. 12] for an introduction.

PDDL-Type Descriptions

Remaining on the simple side of all the above assumptions, a *domain description* contains mainly two parts. First, a specification of a restricted first-order language \mathcal{L} from predicate symbols and constants representing the relations and objects considered relevant in modeling the environment; and, second, a set \mathcal{A} of *action descriptions*. These consist for each action $a \in \mathcal{A}$ of $\text{pre}(a)$, the preconditions, i.e., a set of facts in \mathcal{L} that need to be true to execute a ; and $\text{post}(a)$, the postconditions, i.e., a set of (positive or negative) facts in \mathcal{L} that become true by executing a .

To give a very simple example (taken from [9.27, p. 35]), Table 9.1 shows an *action schema* for moving a robot r between adjacent locations l and m . Action schemata are instantiated in the planning process to ac-

Table 9.1

```
(:action move
:parameters (?r - robot ?l ?m - location)
:precondition (and (adjacent ?l ?m) (at ?r ?l)
(not (occupied ?m)))
:effect (and (at ?r ?m) (occupied ?m)
(not (occupied ?l)) (not (at ?r ?l))))
```

tions occurring in plans, by substituting variables by constant objects.

A specific *planning problem*, given a domain, is then specified by the following components over \mathcal{L} :

- I , the *initial situation*, a set of all facts in \mathcal{L} true of the environment before execution starts
- G , the *goal conditions*, a set of facts in \mathcal{L} to bring about by executing a plan

Plans in this framework are typically *partially ordered* sets of ground (variable-free) actions to be executed in accordance with the ordering.

The *planning domain description language* (PDDL, [9.34]) is the current de facto standard for describing planning domains. In fact, PDDL stands for a whole family of languages, allowing or disallowing features such as argument typing, equality handling, conditional action effects, and some restricted form of FOPL statements. To give an example of a conditional effect, consider that the robot in the move action may or may not be loaded with some container ?c, then we specify as an *additional* effect that ?c would move with the robot:

```
(when (loaded ?r ?c)
      % condition
      (and (at ?c ?m) (not(at ?c ?l))))
      % effect.
```

Beyond such variants, there are extensions to the original PDDL of which PDDL2.1 [9.35] is most notable, allowing information to be expressed for temporal planning, e.g., allowing actions to have durations.

Planning is computationally hard. For propositional variants of PDDL-type problems, i.e., those with a finite \mathcal{L} , plan existence is of course decidable. However, solving planning problems is PSPACE-complete [9.36].

9.3.2 Partial-Order Plan Generation

Many planning systems and their underlying planning algorithms accept the aforementioned restrictive assumptions of information completeness, determinism, instantaneousness, and idleness. In consequence, the definition of a planning problem just given applies. The sort of plan to find, then, is a set of actions whose execution transforms the initial situation into a situation with the goal conditions true. (Goal situations need not be unique.) Finiteness need not be assumed for this planning model, but let us do so for simplicity, meaning that all action instances can be assumed to be ground.

Normally, actions in a plan cannot be executed in arbitrary sequence, but have to obey an ordering, making sure that all preconditions of each action are true at

the time of its execution. This order need not be total or linear, but may be partial or nonlinear. The corresponding type of plan is called a nonlinear or *partial-order plan*, which is, in sum, defined as a pair $\langle A, \prec \rangle$ of a set A of action instances and an ordering relation \prec on A . Unordered actions may be executed in either sequence. With some care in the action modeling, they may also be executed *concurrently*, which, assuming instantaneousness, means within the same atomic *time step*.

For a plan to be a *solution* of the given planning problem, every action $a \in A$ needs to have its preconditions true in every possible execution sequence compatible with \prec , and the goal conditions must be true at the end of any such execution.

Starting from the first partial-order planning approaches in the 1970s and 1980s, culminating in the SNLP (systematic nonlinear planning) algorithm [9.37] and its descendants, algorithms that tried to achieve efficiency by systematic, backward-chaining, least-commitment, relatively reasoning-intensive assembly of partial-order plans were developed. Reference [9.4, Sec. 11.3] provides an introduction and references. This line of work is often called *classical planning*.

Here is the algorithmic idea to generate a solution plan: start with the empty plan containing only the initial facts and the goal conditions; iteratively check any condition occurring in the current plan, whether it is true under all \prec -admissible execution sequences; if some condition c is found open, fix it by inserting a new action generating c as an effect or by ordering an existing action in the plan so that c becomes true at the time needed. This would work efficiently if all actions were independent of each other, but usually they are not: a c once achieved in a plan may be threatened later by some action with $\neg c$ in its effects. Much of the algorithmic or heuristic effort in classical planning systems has gone into preventing or handling subgoal interactions in plans efficiently.

A technique employed in many application systems is using predefined subplans as *macros*, which induce a hierarchical structure in plans, motivating their name *hierarchical task networks* (HTNs). Reference [9.27, Chap. 11] gives an introduction. Formally, a subplan can be expressed as a *task*, i.e., an action with preconditions and postconditions, that can be expanded at planning time into its underlying subplan. Tasks may be layered hierarchically. A plan is unfinished as long as it contains unexpanded tasks. The rationale for using HTNs is that the planning domain modeler knows typically – in fact, in all real applications – recurring subplans that the planner need not reinvent every time from scratch.

Moreover, the hierarchical nature of HTNs allows even large plans to be displayed in a way that is manageable for humans. HTNs have been used for a long time, with good results. SIPE (system for interactive planning and execution monitoring) [9.38] was among the first planners used for real applications. More recently, SHOP (simple hierarchical ordered planner) [9.39] is among the top-performing planners.

Starting with GRAPHPLAN [9.40], the field has focused on a set of techniques for partial-order planning algorithms (dubbed *neoclassical planning* by [9.27]), which deviate from the classical algorithms and keep generating top-performance planners. We will elaborate on two of their ingredients, namely, planning graphs and translation into satisfiability problems.

Much of the performance improvement of recent planners is owed to the formulation of planning as a particular family of search problems, for which powerful heuristics exist. For details, see [9.27, Part III].

Planning Graphs

One of GRAPHPLAN's key ideas is to start plan construction with a preprocessing phase, that:

1. determines a necessary criterion for solvability in n time steps (where n is incremented until some answer is found), and
2. yields a structured representation, based on which an n -time-step partial-order plan can be extracted relatively efficiently, if it does exist.

This structure is the *planning graph* (PG), i. e., a bipartite, layered DAG of ground fact nodes and ground action nodes, where layers of fact nodes and action nodes

occur alternatingly. Figure 9.3 sketches three subsequent layers.

The first layer F_0 consists of all fact nodes to represent completely the initial situation of the problem. The last layer is also a fact layer. An edge connects a fact node with an action node iff the corresponding fact is among the action preconditions. An action node is connected with a fact node iff the fact is in the action effects. In addition to the actions \mathcal{A} of the domain description, PGs contain pseudo-actions serving to copy each and every fact of a fact layer to the next fact layer. (Syntactically, a pseudo-action for f has exactly f as its precondition and its effect.) A PG contains in any layer A_i the action nodes corresponding to all actions that are applicable, given the facts of F_{i-1} , and F_i contains all facts contained in all A_i action postconditions.

For a given finite planning problem, the PG is unique up to node sequence within layers. It can be expanded without search by forward-chaining from the last fact layer, inserting all applicable actions in the next action layer and their effects in the next fact layer. The sets of actions and facts grow monotonically over layers, so PG generation terminates for finite domains.

A necessary condition for solvability of the planning problem is the existence of a fact layer containing all goal facts. If such a layer is found at time step n , PG expansion is suspended and *plan extraction* is attempted, assembling a solution plan backwards. A set of actions is selected from each A_i that generate the facts needed in F_i , starting with the goal facts in F_n and needing the preconditions of the A_i actions in F_{i-1} .

Plan extraction is not guaranteed to succeed, as needed actions may be mutually exclusive in a common action layer – the analog to subgoal interaction in classical planning. So, occurrence of the goal conditions in some F_n is a necessary, but not sufficient, criterion for solvability. In general, search is involved in plan extraction from a PG, yet in a much smaller search space than in classical planning algorithms, which typically outweighs by far the effort for constructing the PG first.

The main drawback of PGs is that they require domains to be finite (i. e., propositional), and, moreover, their languages \mathcal{L} have to be limited in size, as all fact and action ground instances are explicitly constructed for expanding the PG.

Planning as Satisfiability

From the early history of planning, researchers have been tempted to express the problem of plan generation as one of inference in classical logic [9.41]. This might be useful for two reasons. First, existing efficient

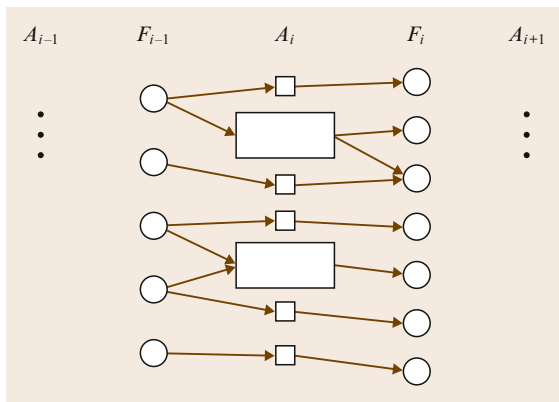


Fig. 9.3 Structure of a planning graph. Circles represent fact nodes, boxes represent action nodes, and small square boxes represent pseudo-operators

logical reasoners might be used for planning, saving the effort of inventing dedicated planning algorithms. Second, logical reasoning within the domain would be seamlessly integrated into planning, obtaining powerful domain representations for free.

Expressing change and inertia efficiently in FOPL has turned out to be problematic due to the *frame problem*; see [9.4, Sec. 10.3] for an introduction. Meanwhile there are several methods to represent planning problems in different logical formalisms, with some *deductive* planners under the top performers of their times. For example, *TALplanner* [9.42] uses a representation in a temporal logic; *Blackbox* [9.43] uses an encoding in propositional logic, its planning algorithm incorporating PGs. A propositional logic encoding offers the option of using an efficient model checker (Sect. 9.1) for inference. This is applicable whenever PGs are. Reference [9.27, Chap. 7] provides an introduction.

A propositional representation of a finite planning problem can be derived relatively schematically from a given domain and problem description, as developed for *Blackbox*. Facts are given an additional situation argument, with actions transiting between situations. The propositional logic representation of initial facts, goal facts, and action effects is straightforward. The interesting part (Sect. 9.2.1) consists of the *explanatory frame axioms*, i. e., a set of axioms formalizing that any change of the truth of a fact between two situations needs to be *explained* by applying an action that has the changed fact values among its effects.

9.3.3 Planning Under Uncertainty

While information completeness, determinism, and idleness are often useful approximations for planning domains at some abstract level, they may not help in other cases. However, that does not mean that all their exact opposites are true (in which case planning would hardly make sense). Many domains show aspects of ignorance, randomness or dynamics, but still offer some degree of information or control. Exploiting this to compensate for any lacking information or control is the point of planning under uncertainty. According to different aspects and grades of imperfection in domain models, it comes in different forms. We sketch one of these forms here and refer to [9.27, P. V] and [9.4, Chaps. 12, 16, 17] for comprehensive introductions. Reference [9.44, P. IV] introduces probabilistic planning techniques in a robotics context. Reference [9.45] develops a unifying view of classical planning as previously described and planning under various forms of uncertainty.

MDPs

Assume that some actions may have nondeterministic effects, and information (which may be approximate or estimated) is available about the relative frequencies of these effects. In this case, representation in a probabilistic framework, such as *Markovian decision processes* (MDPs) is natural. An MDP represents the domain by a finite set of states S , a finite set of actions A , and an action model, which specifies for each $a \in A$ the conditional probability distribution $P_a(s' | s)$ of ending in state s' when executing a in s . For example, assume that the robot's current location is part of the state description; the high-level action of moving from its current location l to an adjacent, free location m (Sect. 9.3.1) produces the standard effect of being at m in 95% of cases; unspecified, maybe unknown, conditions cause the robot to stay at location l in the remaining 5% of cases.

States have associated utilities, where action cost may be modeled as negative utility of the resulting states. Planning in this framework means finding a course of action that takes not only a unique standard outcome for every operator into account, but regards all likely outcomes, including those that are unlikely but would produce very costly effects (think of the possible motion errors of a wheeled platform in the vicinity of a steep staircase).

A plan in this framework is a *policy* as in decision theory, i. e., a function from states into actions. The basic planning algorithms are value iteration (VI) and policy iteration (PI, see [9.4, Chap. 17]), both of which converge to the optimal policy for finite state sets. The difference is that VI does so by approximating the true utility of an optimal policy in a given state, whereas PI finds just the optimal policy, saving the effort of determining its precise utility value.

MDPs still make an assumption that may be inconvenient in some robotic applications, namely that state is observable at execution time, i. e., the agent can precisely determine in which state it is, and can pick the corresponding branch in its policy with certainty. Further skipping the observability assumption leads to *partially observable MDPs* (POMDPs). A POMDP adds to an MDP an observation model consisting of a finite set O of possible observations for the agent to make and of the conditional probabilities $P(o | s)$ of observing o in s . POMDPs are also used as the formal model in mobile robot localization and mapping (see Chap. 37).

Optimal policies for POMDPs are defined in analogy to MDPs, but finding them is computationally hard. A standard approach is to consider the *belief space* rather

than the state space, i.e., the space of probability distributions over the state space, which correspond to the agent's belief of where in state space it probably is after performing an action or an observation. Belief space is fully observable (the agent knows its beliefs with certainty); hence a POMDP can be translated into a belief space MDP and the VI and PI algorithms applied. The belief space is exponentially larger than the state space, and it is continuous. As a consequence, only POMDPs over small state spaces can be effectively handled in this way. Approximation approaches exist, see [9.4, Chap. 17].

9.3.4 Robot Planning

The term *robot planning*, or *plan-based robot control*, in AI refers to the usage of the planning methods just described in the control software of a robot, typically an autonomous mobile robot. It is not meant to replace existing motion or path-planning methods in robotics (Chaps. 5, 26, and 35), but to complement them for more general action planning. There are several purposes for using plans in robot control – be they automatically generated online by the robot or taken from a plan library. Offering a highly granular description of the robot's operation, they can be useful for *optimizing* the overall behavior, for *interacting* with humans as in task-level teleoperation, for *communicating* with fellow robots as in multirobot coordination (Chap. 40), for *learning* complex action structures in highly granular chunks, for *adapting* the behavior to environment changes, and, on a different scale of usefulness, for *engineering* the robot control code by providing a meaningful abstraction level for the programmer. Reference [9.27, Chap. 20] elaborates on these purposes; [9.46] is a collection of research papers spanning them. Reference [9.33, Chap. 5] presents a recent application-oriented perspective on the field.

Corresponding to the multiplicity of purposes and abstraction levels where action planning can be of help,

there is no single method, formalism or approach for robot planning; rather, any of the planning methods can be of use. As a consequence, the term *robot plan* is mostly used in the broad sense formulated in [9.47], namely *that part of the robot's program whose future execution the robot reasons about explicitly*. The specificity of robot planning from the viewpoint of planning methodology comes from the requirement for closed-loop execution and monitoring of whatever plan the robot adopts. Technically, this poses constraints on robot planning, in particular on the domain language, on the action formalism, and on the robot control architecture.

The *domain language* \mathcal{L} for a given planning domain is normally carefully handcrafted, as bad design decisions may make the difference between short and unacceptable planning time of the same planner. For robot planning, there is another issue: the propositions (at least those that need to be observed for plan generation and execution monitoring) need to be effectively determined based solely on sensor data. In general, this is the fundamental *symbol grounding* problem in AI [9.1, 2] (Sect. 9.1). The current pragmatic approach to this is to restrict the dynamic part of the domain language to propositions that can be effectively monitored.

The *action formalism* should ideally support both plan generation and execution monitoring. So it should be expressive enough to resemble, e.g., real-time programming languages, but restricted enough to make reasoning about plans effective. Reference [9.48] discusses these issues and reviews the relatively small range of existing approaches.

Finally, the *robot control architecture* has to make sure that planning and plans are integrated appropriately into the overall robot control. In particular, the advice coming from the recent plan (if any) has to be effectively negotiable with any activity coming from other control components, such as concurrent reactive behaviors; see Chap. 8 for details.

9.4 Robot Learning

Machine learning has been one of the first concerns of AI research, the link between learning and intelligence appearing to be so obvious. One of the first instances of a learning program was Samuel's Checkers program in 1959. The definition of learning is itself as general as the definition of intelligence. It can be basically stated

as *the ability to improve the system's own performance or knowledge based on its experience*. In AI, the main classical learning schemes are based on manipulating symbolic representations to produce new knowledge either by deduction, by induction or by analogy. Statistical learning covers methods that enable the classification

of data, i.e., to deduce models from samples, which is a widespread learning scheme used in robotics and other domains. When the representations are probabilistic, Bayesian reasoning is used. Neural networks, based on formal models of the biological neurons, cover techniques for data classification related to statistical learning.

Another popular learning scheme in robotics is *reinforcement learning*, which is closely related to MDPs and dynamic programming. It is mainly based on a reinforcement signal (a reward) associated to each robot action that is used to reinforce a correct response.

Learning can also be classified as being *supervised* or *unsupervised* depending on whether there is a teacher able to provide the correct answer explicitly after the system's action, i.e., to reward the system, or only the system's own performance is available for this.

This section overviews the most important learning schemes of interest to robotics. Applications of learning in robot control appear throughout this Handbook, e.g., for learning maps in Chap. 37, learning intricate motor coordination patterns in Chap. 60, and evolutionary learning in Chap. 61.

9.4.1 Inductive Logic Learning

One of the main supervised symbolic learning schemes is *inductive logic programming* (ILP) [9.49]. This process is based on logic representations and first-order logic inference, and uses contextual knowledge, i.e., already acquired knowledge. The purpose of ILP is to synthesize (or learn) logic programs given a set of variables for which they return true or false values, and so-called background knowledge. The programs form an explanation of the observations. These programs are of course not unique. Therefore the difficulty is to find a minimal (necessary and sufficient) program that is compatible with the data, i.e., that returns true for the positive variables and false for the negative ones. This program must also be able to generalize, i.e., to produce the correct answer when applied to new instances. In general, ILP is not tractable. Reference [9.4, Sect. 19.5] provides an introduction.

9.4.2 Statistical Learning and Neural Networks

Statistical learning is very widely used in robotics and computer vision. It covers several techniques such as *Bayesian learning*, *kernel methods*, and *neural networks* that classify patterns into categories with the purpose of

learning models from data considered as random variables, the values of which provide a sampling of the model. First there is a training period during which the algorithm is presented with a set of data and associated labels (supervised learning). Then the algorithm will label the data on its own. The purpose of the training phase is to minimize the classification error (or risk or expected loss). We refer to [9.4, Chap. 20] for a general introduction.

Bayesian learning is the application of the Bayes rule to compute the probability of given hypotheses – which are the labels of the classes to which the data belong – given the data. The training period provides the likelihood of the data for each class $P(d|C_i)$. The classification also requires a knowledge of the prior class distribution $P(C_i)$ to apply the Bayes rule. There are several variations of Bayesian learning including *maximum likelihood* (ML) and *expectation maximization* (EM). Bayes techniques have medium computational cost and are easy to implement, but they are not online.

Neural networks (NN) are composed of elementary computational units (neurons) linked by weighted connections. Each unit computes the weighted sum of its inputs and activates or fires if the input is larger than a given threshold (a sigmoid function is often used). Hence this provides for the base of a classifier. The basic structure of a feedforward network, the most commonly used, includes an input layer, one or more hidden layers, and an output layer. Through the connection weights, the output of a NN is a function of its inputs. The training of the NN consists of adjusting the weight to classify the data correctly (supervised learning). Contrary to Bayesian learning, neural networks give no insight into the classifier, but they can be used online.

9.4.3 Reinforcement Learning

Reinforcement learning (RL) is unsupervised learning through acting in the environment, which returns a reward signal. A main reference for RL is [9.50], while [9.4, Chap. 21] gives a general introduction. In RL, the purpose of the robot or the agent is to learn the best *policy*, i.e., sequence of actions to reach a given goal. To do this it has to maximize the cumulated reward. The formal framework of RL is therefore the same as that of MDPs. The robot perceives (in some cases partially) environment states in a set S and is able to accomplish actions in a set A , the sequence of which forms a policy π . There are transition probabilities from a state to the

next upon performing a given action $P(s_{t+1} = s' | s_t, a_t)$. The robot attempts to maximize a value function V expressing the expected cumulated rewards r_t over time. The basic equation expressing the value function and enabling learning of the optimal policy π^* is the Bellman equation

$$V^{\pi^*}(s) = \max_a \left[r(s, a) + \gamma \sum_s P(s' | s, a) V^{\pi^*}(s') \right], \quad (9.1)$$

where $r(s, a)$ is the immediate reward, and the discount factor γ enables future anticipated rewards to be taken into account. A variant to compute the optimal policy is Q-learning [9.51], the idea of which is to learn values of actions rather than those of states.

Most RL work assumes that the action space is discrete, and the application to continuous action spaces is not straightforward given the formulation of the learning process itself. To apply RL to continuous spaces, one usually resorts to sampling.

9.5 Conclusions and Further Reading

From its early days, autonomous robots have been included on the AI research agenda. The first experiments took place in many laboratories in the 1990s, when mobile robot platforms became widely available and affordable. AI's motivation for dealing with robotics is clear: it provides a useful test bed for computational models of agents that include perception, reasoning, and action, as robots allow them to be examined in full integration.

This is similar to AI's *long-term* perspective for robotics: to develop methods and tools that will contribute to closed-loop controllers for smart autonomous robots. But what is the contribution of AI reasoning methods to robotics *today*?

Two lessons have been learned here. First, the described reasoning methods are sufficiently well understood and efficient that they can be applied safely and profitably as ingredients in closed-loop control of robotic agents today. Mid-size league matches in RoboCup soccer are spectacular examples of this; the autonomous control features in the National Aeronautics and Space Administration (NASA) Remote Agent mission [9.52] are another.

Second, the full closed-loop integration of perception, (symbol-based) reasoning, and action would require the symbol-grounding problem [9.1], or a substantial part of it, to be solved. That is, we would have to understand how a robot could perceive its environment in terms of the semantic categories that it uses for reasoning and for planning actions. Understanding the symbol-grounding process fully is among the big problems of AI, and of all of cognitive science for that matter. So robotics had better not wait for AI to find the general solution first. In the meantime, robots can profitably use AI reasoning methods wherever the required data are available in closed-loop control.

To conclude this chapter, let us return to the issue of the robot control architecture to highlight an essential point of functional and physical integration. For the purpose of this overview, we have treated reasoning as a method or set of methods that has to rely on information given by, first, a (human) domain modeler, second, online perception for delivering up-to-date information in symbol form about the current state of the domain, and/or, possibly third, a learner that helps with the recognition or the reasoning itself by generalizing past experience. This view is in fact short-sighted. A fully integrated robot reasoner should help other modules by providing hypotheses, e.g., about what the perception module might currently look for in its sensor data. (*If you tell me you have identified a sink and a dishwasher, then I tell you we might be in a kitchen, and you could watch out for a fridge.*) A two-way integration of reasoning and perception is currently beyond the state of the art in robotics, but is moving into focus. A source of inspiration might be related work from *cognitive vision*, of which [9.53] contains a collection of recent papers.

Further Reading

For a more complete coverage of the methods described here, as well as the of the rest of AI, we refer to AI textbooks, of which the book by Russell and Norvig [9.4] is currently most widely used. Reference [9.5] is to be recommended as a recent introduction specifically to knowledge representation and its issues. Among the sources for first-hand material, let us mention the two journals *Artificial Intelligence* and the *Journal of Artificial Intelligence Research* (JAIR) [9.54], both of which cover the topics addressed here well. The main international conference, the *International Joint Conferences on Artificial Intelligence* (IJCAI), is held biannually every *odd* year.

References

- 9.1 S. Harnad: The symbol grounding problem, *Physica D* **42**, 335–346 (1990)
- 9.2 S. Coradeschi, A. Saffiotti: An introduction to the anchoring problem, *Robot. Auton. Syst.* **43**(2–3), 85–96 (2003)
- 9.3 F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider (Eds.): *The Description Logic Handbook* (Cambridge Univ. Press, Cambridge 2003)
- 9.4 S. Russell, P. Norvig: *Artificial Intelligence: A Modern Approach*, 2nd edn. (Prentice Hall, Englewood Cliffs 2003)
- 9.5 R.J. Brachman, H.J. Levesque: *Knowledge Representation and Reasoning* (Morgan Kaufmann, San Francisco 2004)
- 9.6 W.V.O. Quine: *Methods of Logic*, 4th edn. (Harvard Univ. Press, Cambridge 1955)
- 9.7 Z. Manna, R. Waldinger: *The Deductive Foundations of Computer Programming: A One-Volume Version of "The Logical Basis for Computer Programming"* (Addison-Wesley, Reading 1993)
- 9.8 W. Hodges: Elementary predicate logic. In: *Handbook of Philosophical Logic*, Vol. I, ed. by D. Gabbay, F. Guenther (D. Reidel, Dordrecht 1983)
- 9.9 A. Robinson, A. Voronkov (Eds.): *Handbook of Automated Reasoning* (Elsevier Science, Amsterdam 2001)
- 9.10 M. Davis, G. Logemann, D. Loveland: A machine program for theorem proving, *Commun. ACM* **5**(7), 394–397 (1962)
- 9.11 J. Franco, H. Kautz, H. Kleine Büning, H. v. Maaren, E. Speckenmeyer, B. Selman: Special issue: theory and applications of satisfiability testing, *Ann. Math. Artif. Intell.* **43**, 1–365 (2005)
- 9.12 The Web Ontology Language OWL. <http://www.w3.org/TR/owl-features/>
- 9.13 G. Antoniou, F. v. Harmelen: *A Semantic Web Primer* (MIT Press, Cambridge 2004)
- 9.14 K.L. Chung, F. Ait-Sahila: *Elementary Probability Theory* (Springer, Berlin 2003)
- 9.15 J. Pearl: *Probabilistic Reasoning in Intelligent Systems* (Morgan Kaufmann, San Mateo 1988)
- 9.16 R. Brooks: Intelligence without representation, *Artif. Intell.* **47**, 139–159 (1991)
- 9.17 J. McCarthy, P. Hayes: Some philosophical problems from the standpoint of artificial intelligence, *Machine Intell.* **4**, 463–507 (1969)
- 9.18 H. Levesque, R. Reiter, Y. Lespérance, F. Lin, R. Scherl: Golog: a logic programming language for dynamic domains, *J. Logic Programm.* **31**, 59–83 (1997)
- 9.19 M. Shanahan, M. Witkowski: High-level robot control through logic, *ATAL '00*, 7th Intl. Workshop Intell. Agents VII. Agent Theories Architectures and Languages 2000 (Springer, Berlin 2001) pp.104–121
- 9.20 M. Thielscher: *Reasoning Robots. The Art and Science of Programming Robotic Agents* (Springer, Berlin 2005)
- 9.21 A. Saffiotti, K. Konolige, E.H. Ruspini: A multivalued logic approach to integrating planning and control, *J. Artif. Intell.* **76**, 481–526 (1995)
- 9.22 R.J. Firby: An investigation into reactive planning in complex domains, *AAAI 1987* (Morgan Kaufmann, San Mateo 1987) pp. 202–206
- 9.23 M.P. Georgeff, A.L. Lansky: *Reactive Reasoning and Planning*, *AAAI 1987* (Morgan Kaufmann, San Mateo 1987)
- 9.24 M.P. Georgeff, F.F. Ingrand: Decision-making in an embedded reasoning system, *IJCAI 1989* (Morgan Kaufmann, San Mateo 1989)
- 9.25 M. Boddy, T.L. Dean: Solving time-dependent planning problems, *IJCAI 1989* (Morgan Kaufmann, San Mateo 1989)
- 9.26 S. Zilberstein: Operational rationality through compilation of anytime algorithms, *AI Mag.* **16**(2), 79–80 (1995)
- 9.27 M. Ghallab, D. Nau, P. Traverso: *Automated Planning: Theory and Practice* (Morgan Kaufmann, San Francisco 2004)
- 9.28 R.E. Fikes, N.J. Nilsson: strips: a new approach to theorem proving in problem solving, *J. Artif. Intell.* **2**, 189–208 (1971)
- 9.29 R.E. Fikes, P.E. Hart, N.J. Nilsson: Learning and executing generalized robot plans, *J. Artif. Intell.* **3**, 251–288 (1972)
- 9.30 N.J. Nilsson: Shakey the Robot. SRI International, Tech. Note TN 323, 1984. www.ai.sri.com/shakey/
- 9.31 J. Rintanen, J. Hoffmann: An overview of recent algorithms for AI planning, *KI* **15**(2), 5–11 (2001)
- 9.32 PLANET: European Network of Excellence in AI Planning. <http://www.planet-noe.org/>
- 9.33 PLANET Technological Roadmap on AI Planning and Scheduling. <http://www.planet-noe.org/service/Resources/Roadmap/Roadmap2.pdf>. 2003
- 9.34 D. McDermott, M. Ghallab, A. Howe, A. Ram, M. Veloso, D. S. Weld, D. E. Wilkins: *PDDL – The Planning Domain Definition Language*, Tech Report, Vol. CVC TR-98-003/DCS TR-1165 (Yale Center for Computational Vision and Control, New Haven 1998)
- 9.35 M. Fox, D. Long: PDDL2.1: an extension to PDDL for expressing temporal planning domains, *J. Artif. Intell. Res.* **20**, 61–124 (2003)
- 9.36 T. Bylander: The computational complexity of propositional strips planning, *J. Artif. Intell.* **69**, 165–204 (1994)
- 9.37 D. McAllester, D. Rosenblitt: Systematic nonlinear planning, *AAAI 1991* (Morgan Kaufmann, San Mateo 1991)

- 9.38 D. Wilkins: Domain-independent planning: representation and plan generation, *J. Artif. Intell.* **22**, 269–301 (1984)
- 9.39 D.S. Nau, T.C. Au, O. Ilghami, U. Kuter, M. Murdock, D. Wu, F. Yaman: Shop2: an HTN planning system, *J. Artif. Intell. Res.* **20**, 379–404 (2003)
- 9.40 A.L. Blum, M.L. Furst: Fast planning through plan graph analysis, *J. Artif. Intell.* **90**, 281–300 (1997)
- 9.41 C. Green: Application of theorem proving to problem solving, *IJCAI 1969* (Morgan Kaufmann, San Mateo 1969)
- 9.42 P. Doherty, J. Kvarnström: TALplanner: a temporal logic based planner, *AI Mag.* **22**(3), 95–102 (2001)
- 9.43 H. Kautz, B. Selman: Unifying SAT-based and graph-based planning, *IJCAI*, Stockholm 1999 (Morgan Kaufmann, San Mateo 1999)
- 9.44 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
- 9.45 B. Bonet, H. Geffner: Planning with incomplete information as heuristic search in belief space, *AIPS 2000* (AAAI, Menlo Park 2000)
- 9.46 M. Beetz, J. Hertzberg, M. Ghallab, M.E. Pollack (Eds.): *Advances in Plan-Based Control of Robotic Agents*, Vol. 2466 (Springer, Berlin 2002)
- 9.47 D. McDermott: Robot planning, *AI Mag.* **13**(2), 55–79 (1992)
- 9.48 M. Beetz: Plan representation for robotic agents, *AIPS*, Toulouse 2002 (AAAI, Menlo Park 2002)
- 9.49 N. Lavrac, S. Dzeroski: *Inductive Logic Programming: Techniques and Applications* (Ellis Horwood, New York 1994)
- 9.50 R.S. Sutton, A.G. Barto: *Reinforcement Learning: An Introduction* (MIT Press, Cambridge 1998)
- 9.51 C.J. Watkins: Models of Delayed Reinforcement Learning. Ph.D. Thesis (Cambridge Univ., Cambridge 1989)
- 9.52 N. Muscettola, P. Nayak, B. Pell, B.C. Williams: Remote Agent: to boldly go where no AI system has gone before, *J. Artif. Intell.* **103**, 5–47 (1998)
- 9.53 H.I. Christensen, H.H. Nagel (Eds.): *Cognitive Vision Systems – Sampling the Spectrum of Approaches LNCS* (Springer, Berlin 2006)
- 9.54 *J. Artif. Intell. Res.* <http://www.jair.org>

Robot Structures

Part B

Part B Robot Structures

Ed. by Frank C. Park

10 Performance Evaluation and Design Criteria

Jorge Angeles, Montreal, Canada
Frank C. Park, Seoul, Korea

11 Kinematically Redundant Manipulators

Stefano Chiaverini, Cassino, Italy
Giuseppe Oriolo, Roma, Italy
Ian D. Walker, Clemson, USA

12 Parallel Mechanisms and Robots

Jean-Pierre Merlet, Sophia-Antipolis, France
Clément Gosselin, Quebec, Canada

13 Robots with Flexible Elements

Alessandro De Luca, Roma, Italy
Wayne Book, Atlanta, USA

14 Model Identification

John Hollerbach, Salt Lake City, USA
Wisama Khalil, Nantes, France
Maxime Gautier, Nantes, France

15 Robot Hands

Claudio Melchiorri, Bologna, Italy
Makoto Kaneko, Suita, Osaka, Japan

16 Legged Robots

Shuuji Kajita, Tsukuba, Japan
Bernard Espiau, Saint-Ismer, France

17 Wheeled Robots

Guy Campion, Louvain-la-Neuve, Belgium
Woojin Chung, Seoul, Korea

18 Micro/Nanorobots

Bradley J. Nelson, Zürich, Switzerland
Lixin Dong, Zürich, Switzerland
Fumihito Arai, Sendai, Japan

The chapters contained in **Part B**, Robot Structures, are concerned with the design, modeling, motion planning, and control of the actual physical realizations of a robot. Some of the more obvious mechanical structures that come to mind are arms, legs, and hands; to this list we can add wheeled vehicles and platforms, and robot structures at the micro- and nanoscales. Even for that most basic robotic device, the arm, an incredibly diverse set of structures is possible, depending on the number and types of joints and actuators, and the presence of closed loops in the kinematic structure, or flexibility in the joints and links. Constructing models, and planning and control algorithms for these diverse structures represents an even greater set of challenges.

The topics addressed in these chapters are essential to creating not only the physical robot itself, but also to creating and controlling movements, and manipulating objects in desired ways. As such the connections with the chapters on Robot Foundations (Part A) — particularly the chapters on Kinematics (Chap. 1), Dynamics (Chap. 2), and Mechanisms and Actuation (Chap. 3) — are self-evident. What ultimately distinguishes robotics from other disciplines that study intelligence is that, by definition, robots require a physical manifestation, and by extension must physically interact with the environment. In this regard the topics addressed in these chapters can be said to constitute the most basic layer of this endeavor.

Just as it is difficult to examine human intelligence from a purely abstract perspective, remotely detached from the physical body, so it is difficult to separate the contents of the remaining parts without including in the discussion the actual medium of interaction with the physical world, the (physical) robots themselves. For example, the question of how to coordinate sensing and perception with action (Part C), how to grasp and manipulate objects (Part D), and how to teach multiple robots to cooperate (Part E), must inevitably consider the physical structure of the robot. Robots specialized to various applications and environments (Part F), particularly those intended for direct interaction with humans (Part G), naturally must also consider the robot's physical structure.

With this overview of Part B, we now provide a brief synopsis of each chapter.

Chapter 10, Performance Evaluation and Design Criteria, provides a concise overview of the robot design process, and surveys some of the criteria and tools used in the mechanical design and performance evaluation of robots. Criteria such as workspace volume, local and global dexterity, and elastostatic and elastodynamic

performance are not only applicable to determining the topological structure and physical dimensions of the robot, but can also be useful for, e.g., workpiece placement and kinematic redundancy resolution.

Chapter 11, Kinematically Redundant Manipulators, addresses the motion generation and control of manipulators with redundant kinematic degrees of freedom. Kinematic redundancy affords a robot with an increased level of dexterity that may be used to, e.g., avoid singularities, joint limits and workspace obstacles, and also to minimize joint torques, energy, or other suitable performance criteria. The chapter discusses inverse kinematic redundancy resolution schemes for manipulators ranging from those with just a few degrees of kinematic redundancy, to hyperredundant manipulators that can be kinematically modeled as continuous curves.

Chapter 12, Parallel Mechanisms and Robots, presents an introduction to the kinematics and dynamics of parallel mechanisms such as the well-known Stewart–Gough platform. Parallel mechanisms contain closed loops in their kinematic structure, and as such methods for their analysis differ considerably from those for their serial counterparts. This chapter discusses topics ranging from type synthesis and forward and inverse kinematic solutions of parallel mechanisms, to an investigation of their singularity behavior, workspace characterization, static and dynamic analysis, and practical issues in their design.

Chapter 13, Robots with Flexible Elements, addresses the dynamic modeling and control of robots with flexibility in the joints and links. Because the control methods developed to compensate for joint versus link flexibility are structurally different, the chapter is organized such that these two types of flexibility are examined independently. The methods, however, can be extended to the case when both joint and link flexibilities are present, possibly even dynamically interacting at the same time. The chapter also examines the typical sources of flexibility in industrial robots.

Chapter 14, Model Identification, discusses methods for determining the kinematic and inertial parameters of robot manipulators. For kinematic calibration, the primary aim is to identify the geometric Denavit–Hartenberg parameters or equivalent, typically by sensing a combination of joint and endpoint positions. Inertial parameters in turn are estimated through the execution of a trajectory while additionally sensing one or more components of joint forces or torques. The chapter is organized such that both kinematic and inertial parameter identification are cast into a common framework of

least-squares parameter estimation: common features relating to the identifiability of the parameters, adequacy of the measurement sets, and numerical robustness are identified and emphasized for both the kinematic and inertial parameters.

Chapter 15, Robot Hands, investigates the principal issues behind the design, modeling, and control of robot hands. Beginning with a discussion of levels of anthropomorphism, and the characterization of robot hand dexterity, the chapter investigates the relevant design issues for robot hands, actuation and transmission architectures, and available sensing technologies. The dynamic modeling and control of robot hands are made challenging not only by the complex kinematic structure, but also by the flexible transmission elements, and the chapter devotes particular attention to these issues.

Chapter 16, Legged Robots, discusses the myriad issues involved in the design, analysis, and control of legged robots. Beginning with a history of legged robot research, the chapter provides an analysis of cyclic walking, and the control of biped robots based on the forward dynamics and the zero-moment point (ZMP). Multi-legged robots, such as dynamic quadrupeds inspired by mammals and behavior-based multilegged robots, are also discussed, as are hybrid leg-wheel-arm robots,

tethered walking robots, and even legged robots capable of climbing walls.

Chapter 17, Wheeled Robots, provides a general and comprehensive description of wheeled mobile robots. The chapter begins with a discussion of robot mobility based on the types of wheels and the nature of the kinematic constraints, followed by a discussion of kinematic and dynamic state space models, and structural properties of wheeled robot models such as controllability, nonholonomy, and stabilizability. The chapter also discusses feedback linearizability in the context of nonlinear control, and concludes with a detailed classification of the possible robot structures based on the number and types of wheels.

Chapter 18, Micro/Nanorobots, provides an overview of the state of the art in micro- and nanorobotics. The former entails robotic manipulation of objects with dimensions in the millimeter to micron range, as well as the design and fabrication of autonomous robotic agents within this size range (nanorobotics is defined in the same way, but for dimensions smaller than a micron). The chapter outlines scaling effects, actuation, and sensing and fabrication at these scales, and also applications to microassembly, biotechnology, and the construction and characterization of micro- and nano-electromechanical systems.

10. Performance Evaluation and Design Criteria

Jorge Angeles, Frank C. Park

This chapter is devoted to the design of robots, with a focus on serial architectures. In this regard, we start by proposing a stepwise design procedure; then, we recall the main issues in robot design. These issues pertain to workspace geometry, the kinetostatic, the dynamic, the elastostatic, and elastodynamic performance. In doing this, the mathematics behind the concepts addressed is briefly outlined to make the chapter self-contained.

We survey some of the tools and criteria used in the mechanical design and performance evaluation of robots. Our focus is limited to robots that are (a) primarily intended for manipulation tasks and (b) supplied with serial kinematic chains. The kinematics of parallel robots is addressed in detail in Chap. 12. Wheeled robots, walking robots, multifingered hands, and other similar specialized structures are studied in their own chapters.

10.1	The Robot Design Process	229
10.2	Workspace Criteria	231
10.2.1	Reaching a Set of Goal Frames	233
10.2.2	Workspace Volume and Topology	233
10.3	Dexterity Indices	235
10.3.1	Local Dexterity for Open Chains	235
10.3.2	Dynamics-Based Local Performance Evaluation	237
10.3.3	Global Dexterity Measures	237
10.3.4	Closed-Chain Dexterity Indices	238
10.3.5	Alternative Dexterity-Like Measures	238
10.4	Other Performance Indices	238
10.4.1	Acceleration Radius	238
10.4.2	Elastostatic Performance	239
10.4.3	Elastodynamic Performance	241
	References	242

The most obvious application of the criteria and tools described in this chapter is in the mechanical design of a robot. Robot design differs from the design of single-degree-of-freedom machinery in that the latter is intended for one specific task, e.g., picking up a workpiece from a belt conveyor and placing it on a magazine. Moreover, the conveyor is synchronized with the manipulating machine and the magazine is stationary, with well-defined locations where each workpiece is to be placed. Manipulation robots, in contrast, are not in-

tended for one specific task, but rather for a *family of tasks* falling within one class of workpiece motions, e.g., planar, spherical, translational, or motions produced by systems of the Selective Compliance Assembly Robot Arm (SCARA) type, also known as *Schönflies displacements* [10.1]. The challenge that robot designers face is therefore one of *uncertainty* in the specific task that the robot will be required to execute. Design criteria have been devised to help the designer cope with uncertainty, as discussed herein.

10.1 The Robot Design Process

Given a family of tasks that constitute the *functional requirements* in the design process, besides more-detailed *design specifications*, the role of the designer consists in

producing a robot that will meet all the requirements and specifications. The various stages in the robot design job at hand are intended to:

1. determine the *topology* of the kinematic chain underlying the mechanical structure. Under this item we consider first the robot type: serial, parallel or hybrid. Then, a decision is to be made on the layout of the various *subchains* in terms of the type of joints, most commonly, revolute and prismatic. Recently, one additional type has been recognized to be equally useful, the Π -joint, coupling two links under relative translation by means of two other links undergoing identical angular displacements, although about different parallel axes. The four links form a parallelogram four-bar linkage [10.2];
2. determine the *geometric dimensions* of the various links defining the *robotic architecture*, as required to fill a table of *Denavit–Hartenberg parameters* [10.3] so as to satisfy workspace requirements. Although these parameters are usually understood to include the joint variables, these variables do not affect the *robot architecture*; they determine instead the *robot posture*;
3. determine the structural dimensioning of the various links and joints, as needed to meet static load requirements, where load includes both forces and moments – wrenches – under either the most demanding or the most likely operation conditions, depending on the design philosophy adopted at the outset;
4. determine the structural dimensioning of the various links and joints, as needed to meet dynamic load requirements, where loads are inertia effects of links and manipulated object;
5. determine the elastodynamic dimensioning of the overall mechanical structure, including the actuator dynamics, to avoid a specific spectrum of excitation frequencies under either the most demanding or the most likely operation conditions;
6. select the actuators and their mechanical transmissions for the operation conditions adopted at the outset to cope with task uncertainty.

The above stages can be performed sequentially, in the order given above: (i) first, the topology is determined based on the family of tasks specified at the outset and the *shape* of the workspace, as discussed in Sect. 10.2.2; (ii) the link geometry is defined based on the workspace requirements, which include the *maximum reach*, and the topology defined in stage 1; (iii) with the link geometry thus defined, the structural dimensioning of links and joints (unless the robot under design is parallel, which does not fall within the scope of this chapter, all joints are actuated) is undertaken, so as to support the static loads assumed at the outset; (iv)

with the links and joints dimensioned for static-load conditions, the link centers of mass and link inertia matrices are determined for a preliminary evaluation of the motor torque requirements (this evaluation is preliminary in that it does not consider the dynamic load brought about by the actuators; this load can be significant, even in the case of parallel robots, which can have all their motors fixed to the robot base); (v) with the links assumed rigid, joint stiffness is assumed, based on experience or using data from a similar robot, which then leads to an elastodynamic model whose natural modes and frequencies can be determined at a selected set of robot postures (dynamic behavior of the structure is dependent on robot posture) by means of scientific code such as Matlab or computer-aided engineering (CAE) code such as Pro/Engineer or ANSYS; and (vi) if the frequency spectrum of the robot structure is acceptable, the designer can continue to motor selection; otherwise, a redimensioning is required, which means returning to stage 3.

Even though a design cycle can be completed as outlined above, the designer must now incorporate into the elastodynamic model the structural and inertial data provided by the motor manufacturer. This requires a return to stage 5 and a new elastodynamic analysis. It is thus apparent that the robot design process has one element in common with engineering design in general: both are iterative and open-ended [10.4]. Remarkably, however, the various items driving each design stage are, to a large extent, independent of each other, e.g., topology and geometry can be determined independently from motor selection. Obviously, all issues interact in the overall design process, but, within certain design specifications, the various items do not contradict each other, as to warrant a multiobjective design approach. That is, the optimum design of serial robots can be accomplished fairly well by means of a sequence of single-objective optimization jobs. Again, the results of the last stage, motor selection, must be integrated into an overall mathematical model to test the overall performance. One reference addressing practical optimization issues in the conceptual design of industrial robots is [10.5].

Only when the physical limits of components have been exhausted may a radical redesign requiring a return to stage 1 be warranted. This is the case with SCARA systems. Current industrial topologies of these robots are usually of the serial type, with some exceptions, like the *Konig* and *Hartman* RP-AH series robots with parallel architecture [10.6], which feature two serial SCARA systems sharing one common end-effector. The quest for shorter cycle times, as for an industry test cycle (see Sect. 10.2.1), has prompted the industry to look for

alternatives to serial architectures. This is how ABB Robotics is currently marketing a parallel robot, the *FlexPicker*, built upon *Clavel's* Delta robot [10.7], to which a fourth axis has been added in series with the first three. The latter are laid out in a symmetric, parallel architecture that enables Delta to produce pure translations of its moving platform. The shortest cycle time reported by Adept Technology is 420 ms for a payload of 2 kg (with the Adept Cobra s600, a serial robot) but other manufacturers claim even shorter times.

This chapter is organized according to the various stages of the robot design process outlined earlier. Noting that topology selection and geometric dimensioning are tightly coupled in the kinematic design process, we

first begin with an examination of workspace criteria: we review methods for determining the topology of the kinematic chain, followed by the geometric dimensions so as to satisfy workspace requirements. We then review in detail the various criteria developed for characterizing a robot's manipulating capability, focusing on quantitative notions of dexterity based on both kinematic and dynamic models. We then examine methods for structural dimensioning of the links and joints so as to meet both static and dynamic load requirements. Finally, we discuss elastodynamic dimensioning, and actuator and gear sizing, taking into account properties such as the natural frequency of the robot, and force and acceleration capability requirements.

10.2 Workspace Criteria

The most obvious consideration in designing a robot is that its workspace has a set of *required characteristics*. This is a fundamental problem in classical mechanism design, and raises the obvious question of how a user can specify those characteristics.

Issues to consider here pertain, mostly, to what Vijaykumar et al. [10.8] termed the *regional structure* of a manipulator. This applies to manipulators with a *decoupled architecture*, whose last three revolute have concurrent axes, thereby forming a *spherical wrist*, the point of concurrency being the *wrist center*. The manipulation task of architectures of this kind thus allows for a *decoupling* of the positioning and the orientation subtasks: the regional structure, consisting of the first three joints, is first postured so as to locate the center of its wrist at a specified point $C(x, y, z)$; then, the *local structure*, i.e., the wrist, is postured so as to make the end-effector (EE) attain a specified orientation with respect to a frame fixed to the base, given by a rotation matrix.

Most algorithms reported in the literature to determine the workspace of a given robot refer to the workspace of the regional structure. Here, we should distinguish between the workspace of the kinematic chain, regardless of the physical implementation of the chain, and that of the physical robot. In the former, all revolute joints are capable of unlimited rotations about their axes; in the latter, joint limits are needed, for example, to avoid wire entanglement. In the early stages of robot design, joint limits need not be considered, the workspace thus exhibiting symmetries that are proper of the type of joints of the regional structure. If the first joint is a revo-

lute, the workspace has an axis of symmetry, namely, the axis of this revolute joint; if the first joint is prismatic, the workspace has an extrusion symmetry, with the direction of extrusion given by the direction of motion of this joint. As prismatic joints are infinitely extensive, so is the kinematic workspace of a robot with a prismatic joint. The kinematic workspaces of robots with prismatic joints are usually displayed for a finite portion of this workspace.

In the case of parallel robots, to be studied in full detail in Chap. 14, the regional structure is elusive, in general. The usual practice when displaying the workspace for these robots is to assume a constant orientation of the moving plate, the counterpart of the EE of serial robots [10.9]. A common architecture of parallel robots, which arises quite naturally in the design process, entails identical legs symmetrically placed both on the base platform and on the moving platform. Each leg is, in turn, a serial kinematic chain with one or two active joints, all others being passive. The workspace of this kind of robots also exhibits certain symmetries, but no axial symmetry. The symmetries are dictated by the number of legs and the types of actuated joints.

Coming back to serial robots, the workspace can be defined by an envelope that is essentially of one of two types, either a manifold or a surface that is smooth *almost everywhere*, i.e., smooth everywhere except for a set of points of *measure zero* in the Lebesgue sense [10.10]. Broadly speaking, a set of measure zero on a surface is a curve, e.g., a meridian on a sphere, or a set of isolated points on a line, e.g., the set of rational numbers on the real line. A paradigm for this second kind of workspace

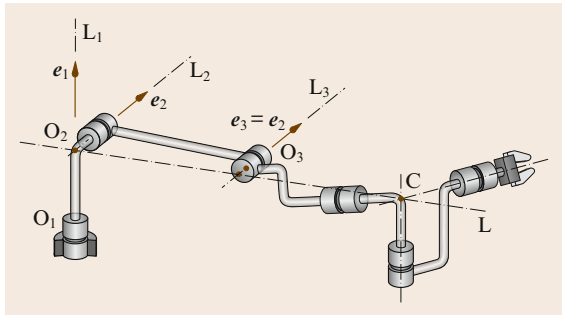


Fig. 10.1 A Puma robot in a fully stretched posture (after [10.11])

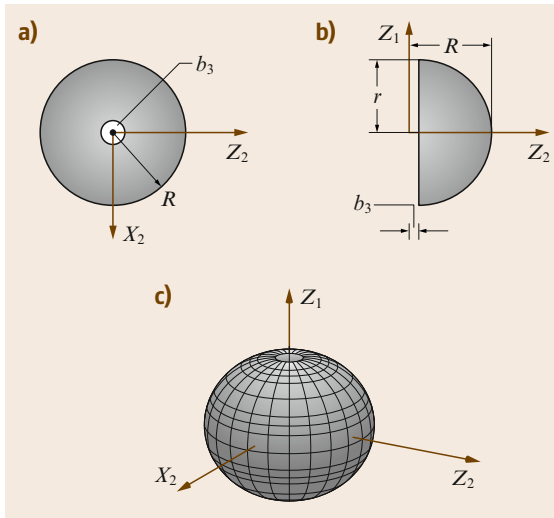


Fig. 10.2a-c The workspace of a Puma robot (after [10.11])

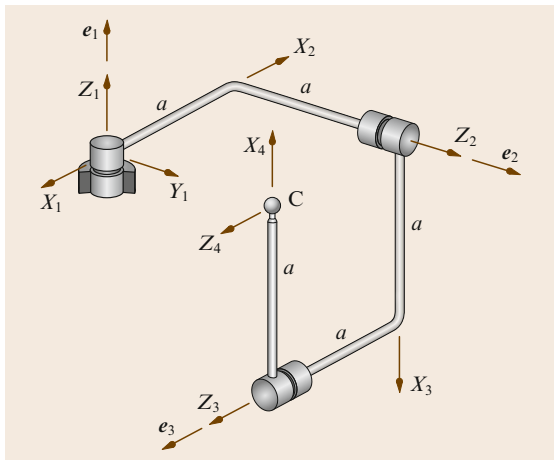


Fig. 10.3 An orthogonal three-revolute robot (after [10.11])

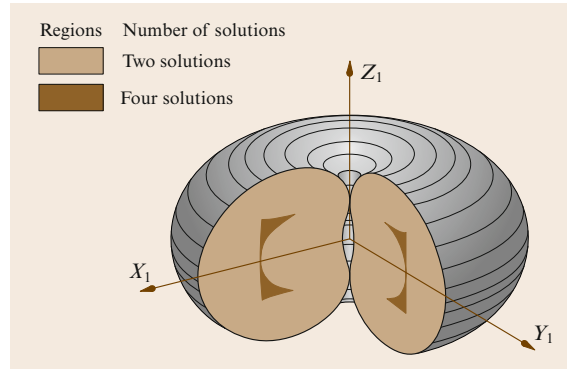


Fig. 10.4 The workspace of the orthogonal robot of Fig. 10.3 (after [10.11])

is that of the Puma robot, whose kinematic chain is displayed in Fig. 10.1. In this figure, the regional and the local structures are clearly distinguished, the former being *fully extended*. The workspace of this robot is obtained upon locking all joints but the second, when the robot is in the posture shown in Fig. 10.1. Then, the second joint is fully rotated about its axis, the center C of the wrist then describing a circle of radius R equal to the distance of C from the line L_2 , the plane of the circle being normal to this line and lying a distance b_3 from the axis L_1 of the first joint. This distance is known as the *shoulder offset*. Now, with all joints locked again, but this time with the first joint unlocked, the robot is turned as a rigid body about L_1 . The result is the toroid of Fig. 10.2. Notice that the solid enclosed by this surface is the result of the Boolean operation $\mathcal{S} - \mathcal{C}$, where \mathcal{S} is the sphere of radius R centered at point O_2 of Fig. 10.1, while \mathcal{C} is the infinite cylinder of radius b_3 and axis L_1 , which appears as Z_1 in Fig. 10.2. It is noteworthy that, although this workspace can be readily generated by a simple Boolean operation, it cannot possibly be generated by an implicit function of the form $f(x, y, z) = 0$ because the surface is not a manifold.

Robots with manifold workspaces are not common in industry. We display in Fig. 10.3 an architecture for the regional structure of a six-axis decoupled robot, with its neighboring axes mutually orthogonal and at the same distance a from each other. The common normals to the two pairs of axes, X_2 and X_3 , also lie at the same distance a , as do X_4 from X_3 and C from Z_3 . Point C is the center of the spherical wrist, the latter not being included in the figure. The workspace of this robot admits a representation of the form $f(x, y, z) = 0$ [10.11], which produces the manifold workspace of Fig. 10.2. The shaded internal region

of the workspace includes all points admitting four real inverse-kinematics solutions, all other points admitting only two.

Given that any point of the workspace boundary represents a positioning singularity – different from an orientation singularity – manipulators with workspace boundaries that are not manifolds exhibit double singularities at the edges of their workspace boundary, which means that at edge points the rank of the robot Jacobian becomes deficient by two. At any other point of the workspace boundary the rank deficiency is by one.

Design rules based on the *shape* of the workspace can now be drawn.

1. If the workspace required is axially symmetric and finite, use a serial robot with a regional structure composed of revolute joints only.
2. If the workspace required is prismatic and *infinite*, use a serial robot with regional structure having one first joint of the prismatic type. Here, *infinite* is used in a restricted sense, meaning much larger in one direction than the others. Moreover,
 - if one direction is required to be much larger than the others, then practical implementations of prismatic joints are available in the form of rails either overhead, thereby giving rise to *gantry* robots, or on the floor.
 - if two directions are required to be much larger than the other, then use a wheeled mobile robot carrying a manipulator on top. A famous realization of this concept is the National Aeronautical and Space Agency's (NASA) *Soyjourner* used in the *Pathfinder* mission to Mars in 1997.
3. If axial symmetry is not required, but rather a workspace with various coplanar axes of symmetry, similar to those of regular polygons, use a parallel robot.

10.2.1 Reaching a Set of Goal Frames

Closely related to the problem of workspace specification is that of task specification. In mechanism design it is customary to specify a set of coordinate frames in space, and to design a mechanism with an a priori specified topology that can visit these frames. An order in which the frames must be reached may be given. In the event that not all of the frames are reachable, then one may seek a mechanism that comes *closest*, in some suitable sense, to the specified frames. The lit-

erature on this classical mechanism design problem is vast – see, e.g., [10.1, 12, 13] and the references cited therein. Some further remarks in connection with this goal-frame approach to robot dimensioning are noteworthy.

1. Reaching *exactly* the desired frames may not always be desired or possible: in some cases it is better to use an optimization approach that allows for solutions that will visit the desired poses within a minimum error (provided that an error norm can be suitably engineered, of course).
2. It has been claimed [10.9] that interval analysis allows not only a discrete set of desired poses but also a full six-dimensional (6-D) workspace to be met while taking into account manufacturing errors.
3. The branching problem occurring in single-degree-of-freedom mechanisms may also occur in robot design: a design solution based on via points may indeed visit the prescribed poses, but not all of these may be reachable within the same assembly mode. This problem is exacerbated in the design of serial robots, as a six-degree-of-freedom, revolute-coupled robot may admit up to 16 distinct postures – branches – for one given EE pose [10.14, 15].
4. While a robot designed to visit a set of prescribed poses via its end-effector will be able to visit that set, we should not forget that the purpose of using robots is first and foremost to be able to perform not one single task, but rather a family of tasks. In this light, the set of poses for which a robot is designed might as well be a task that is *representative* of that family.

In connection with remark 4 above, we can cite the design or evaluation of **SCARA** systems. A **SCARA** system is a four-degree-of-freedom serial robot capable of tasks that lie within the *Schönflies* subgroup of the group of rigid-body displacements [10.16, 17], namely the set of three-dimensional displacements augmented with a rotation about an axis of fixed direction. In these systems, the task at hand is given by two vertical segments joined by one horizontal segment. Moreover, the length of the vertical segments is 25.0 mm, that of the horizontal segment being 300.0 mm. While the end-effector is traversing the horizontal segment, moreover, it should rotate about a vertical axis through an angle of 180°. This task specification, which has been adopted by **SCARA** manufacturers, does not indicate how to negotiate the corners, which is left to the imagination of the robotics engineer.

10.2.2 Workspace Volume and Topology

Reachable and Dexterous Workspace

Beginning with the early work of Roth [10.18], there have been many studies on the relationship between manipulator kinematic geometry and its workspace. Most studies have focused on a classification of the workspace into two components, the *reachable* and the *dexterous* workspace [10.19]. Given a reference point P attached to a manipulator end-effector, such as the center of the spherical wrist, or a point on the EE, the reachable workspace is defined to be the set of points in physical space that can be reached by P . The dexterous workspace, on the other hand, is the set of points that can be reached by P with arbitrary EE orientations.

The early literature on workspace focuses on numerical and algebraic methods to characterize these workspaces. Reachable and dexterous workspaces have been analyzed using numerical techniques by Kumar and Waldron [10.19], Yang and Lee [10.20], and Tsai and Soni [10.21], among others. The advantage of these schemes over algebraic approaches is that kinematic constraints can be readily included. More-general design principles or insights, however, are more difficult to come by using these techniques. Among the algebraic approaches to workspace characterization, a topological analysis of robot workspace is given by Gupta and Roth [10.22] and Gupta [10.23]; here the concept of workspace holes and voids is defined, and conditions for their existence are identified. The shape of the reachable and dexterous workspaces is also analyzed as a function of P .

Further studies of workspace analysis were reported by Freudenstein and Primrose [10.24] and by Lin [10.25], where precise relationships between kinematic and workspace parameters are developed, and a class of three-joint manipulators is optimized for workspace volume. A more general analysis of workspace optimization is given in Vijaykumar et al. [10.8]. Performance criteria for manipulators are defined here in terms of the dexterous workspace; given that a manipulator satisfies certain constraints on its Denavit–Hartenberg parameters, it is shown that the optimal 6R design is the elbow manipulator.

A typical design of robot regional architecture is of the *orthogonal type*, consisting of one revolute of the vertical axis and two revolute of the horizontal axes, one of which intersects the vertical axis. Moreover, the most common architecture includes intermediate and distal links of identical lengths. The workspace

of this architecture is thus a sphere of radius equal to twice that common link length. The volume of this workspace is thus determined by the length in question. As shown by Yoshikawa [10.26], the workspace of the two-link planar manipulator defined by the last two links of the foregoing regional architecture is of maximum area for a prescribed reach and equal link lengths. As a result, the volume of the same regional architecture is similarly of maximum volume for a prescribed reach.

Differential-Geometric Workspace Characterization

Workspace can also be approached from a differential-geometric perspective, by regarding the configuration space of a robotic end-effector frame as a subset of the special Euclidean group $SE(3)$. An important physical consideration in defining the workspace volume of spatial mechanisms is that it should not depend on the choice of fixed reference frame. Less obvious but just as important is the requirement that the volume should not depend on which point of the last link the end-effector frame is fixed to. This last condition has the following physical significance: if the EE were enlarged or shrunk, then the robot would have the same workspace volume. The workspace volume of a robot therefore depends only on the joint axes.

The workspace volume of a robot is defined by regarding $SE(3)$ as a Riemannian manifold, so that the workspace volume is simply the volume of the image of the forward kinematic map f with respect to the volume form on $SE(3)$. It is known that $SE(3)$ has a bi-invariant volume form, that is, the notion of volume is invariant with respect to the choice of both the fixed (base) and moving (end-effector) frames – see, e.g., Loncaric [10.27]. Paden and Sastry [10.28] provide the following visualization for this volume form: Suppose an airplane is restricted to move within a cube of airspace of length 1 km on a side. At each point within this cube the airplane can point itself anywhere in a 4π solid angle and roll 2π about the direction it is pointing. The orientation volume at such a point is $4\pi \times 2\pi = 8\pi^2 \text{ rad}^3$. Multiplying by the positional volume one obtains $8\pi^2 \text{ rad}^3 \text{ km}^3$ for the volume of the free configuration space of the aircraft.

This depiction is the notion of workspace volume used for robots; it has the advantage of being able to trade off orientation freedom for positional freedom smoothly, unlike the popular notion of dexterous workspace. Note that the actual numerical value one obtains will depend on the choice of length scale for physical space; this

in itself does not pose a serious problem for workspace volumes, as long as the same length scale is maintained when comparing different workspaces.

In [10.28] *Paden* and *Sastry* show that the optimal 6R manipulator that maximizes the workspace volume subject to a kinematic length constraint is the

elbow manipulator. This result is consistent with the earlier finding of *Vijaykumar* et al. [10.8], but the authors employ the geometric framework outlined above. Moreover, the results are obtained without some of the a priori assumptions on the kinematic structure made by *Vijaykumar*.

10.3 Dexterity Indices

10.3.1 Local Dexterity for Open Chains

Dexterity can be defined as the ability to move and apply forces and torques in arbitrary directions with equal ease, the concept thus belonging to the realm of *kinetostatics*, which is the study of the interplay between *feasible twists* and *constraint wrenches* in multibody mechanical systems under *static conservative conditions*. Here, twist is the six-dimensional array of velocity variables of a rigid body, involving three components of a landmark-point velocity and three of angular velocity; wrench, in turn, is the six-dimensional array of static variables acting on a rigid body, three accounting for the resultant force applied at the same landmark point and three for the concomitant moment acting on the same body.

Salisbury and *Craig* [10.29] introduced the concept of dexterity when working on the design of articulated hands. At issue is the way in which input joint velocity errors propagate to the output velocities of each fingertip. To illustrate this concept, let $\mathbf{J}(\boldsymbol{\theta})$ denote the Jacobian of the forward kinematic map, i. e.,

$$\mathbf{t} = \mathbf{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \quad (10.1)$$

in which $\boldsymbol{\theta}$ and $\dot{\boldsymbol{\theta}}$ denote the vectors of joint variables and joint rates, respectively, while \mathbf{t} is the EE *twist*, defined, in turn, as,

$$\mathbf{t} = \begin{pmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{pmatrix} \quad (10.2)$$

with $\boldsymbol{\omega}$ denoting the angular velocity of the EE and $\dot{\mathbf{p}}$ the velocity of the *operation point* P of the EE, at which the task is specified.

Let n and m be the dimensions of the joint space \mathcal{J} and of the end-effector configuration space \mathcal{G} , respectively, where $n \geq m$. Now, the image under $\mathbf{J}(\boldsymbol{\theta})$ of the unit hypersphere $\{\boldsymbol{\theta} \mid \|\dot{\boldsymbol{\theta}}\| = 1\}$ is an ellipsoid in the space of twists \mathbf{t} . Indeed, if we assume for concreteness that $n = m$ – the more general case $n \neq m$ can also be accommodated, but we refrain from including it here in

order to streamline the discussion that follows – then the *polar decomposition* [10.30] of \mathbf{J} takes the form

$$\mathbf{J} = \mathbf{R}\mathbf{U} = \mathbf{V}\mathbf{R}, \quad (10.3)$$

where \mathbf{R} is an orthogonal matrix, whether proper or improper – if proper, then \mathbf{R} represents a rotation; if improper, then a reflection – while \mathbf{U} and \mathbf{V} are symmetric and at least positive-semidefinite matrices. Moreover, if \mathbf{J} is nonsingular, then \mathbf{U} and \mathbf{V} are both positive-definite and the decomposition is *unique*. In any event, these two matrices are related by a similarity transformation, namely,

$$\mathbf{V} = \mathbf{R}\mathbf{U}\mathbf{R}^\top \quad (10.4)$$

which means that the two matrices have identical real nonnegative eigenvalues. These eigenvalues are nothing but the *singular values* of \mathbf{J} . Indeed, if $\boldsymbol{\Sigma}$ denotes the diagonal representation of \mathbf{U} , with its i -th diagonal entry σ_i denoting the i -th eigenvalue of \mathbf{U} , or of \mathbf{V} for that matter, then the *eigenvalue decomposition* of \mathbf{U} is

$$\mathbf{U} = \mathbf{E}\boldsymbol{\Sigma}\mathbf{E}^\top \quad (10.5)$$

in which \mathbf{E} is the orthogonal matrix whose i -th column \mathbf{e}_i represents the i -th eigenvector of \mathbf{U} , while $\mathbf{R}\mathbf{e}_i$ denotes the i -th eigenvector of \mathbf{V} . Now, if we substitute the above decomposition of \mathbf{U} into the polar decomposition of \mathbf{J} , (10.3), we obtain

$$\mathbf{J} = \mathbf{R}\mathbf{E}\boldsymbol{\Sigma}\mathbf{E}^\top \quad (10.6)$$

which is nothing but the *singular-value decomposition* of \mathbf{J} , the diagonal entries of $\boldsymbol{\Sigma}$ being the *singular values* of \mathbf{J} .

Now we can provide a geometric interpretation of the forward kinematic mapping of (10.1), for we can rewrite this mapping in the form

$$\mathbf{t} = \mathbf{R}\mathbf{U}\dot{\boldsymbol{\theta}}. \quad (10.7)$$

At nonsingular postures, the Jacobian is invertible, and hence so is \mathbf{U} , whence the foregoing expression leads to

$$\dot{\boldsymbol{\theta}} = \mathbf{U}^{-1}\mathbf{R}^\top\mathbf{t}. \quad (10.8)$$

Furthermore, if we assume that all the components of both the twist array \mathbf{t} and the joint-rate array $\dot{\boldsymbol{\theta}}$ have the same physical units, which is the case for purely positioning or purely orienting manipulators with only revolute joints, then we can take the Euclidean norm of both sides of (10.8), thereby obtaining

$$\|\dot{\boldsymbol{\theta}}\|^2 = \mathbf{t}^\top \mathbf{R} \mathbf{U}^{-2} \mathbf{R}^\top \mathbf{t}. \quad (10.9)$$

Moreover, if we substitute \mathbf{U} by its eigenvalue decomposition, (10.5), in the above equation, we obtain

$$\|\dot{\boldsymbol{\theta}}\|^2 = \mathbf{t}^\top \mathbf{R} \mathbf{E} \boldsymbol{\Sigma}^{-2} \mathbf{E}^\top \mathbf{R}^\top \mathbf{t}$$

and, if we let

$$\mathbf{v} = \mathbf{E}^\top \mathbf{R}^\top \mathbf{t} \quad (10.10)$$

then the above expression for $\|\dot{\boldsymbol{\theta}}\|^2$ becomes

$$\mathbf{v}^\top \boldsymbol{\Sigma}^{-2} \mathbf{v} = \|\dot{\boldsymbol{\theta}}\|^2. \quad (10.11)$$

Now, if the i -th component of \mathbf{v} is denoted by v_i , for $i = 1, \dots, n$, and we look at the mapping of the unit ball in \mathcal{J} , $\|\dot{\boldsymbol{\theta}}\|^2 = 1$, (10.11) leads to

$$\frac{v_1^2}{\sigma_1^2} + \frac{v_2^2}{\sigma_2^2} + \dots + \frac{v_n^2}{\sigma_n^2} = 1 \quad (10.12)$$

which is the canonical equation of an ellipsoid of semi-axes $\{\sigma_i\}_1^n$ in the \mathcal{J} -space, i. e., the space of Cartesian velocities, or twists of the EE. Notice that the ellipsoid in question takes its canonical form when represented in a coordinate frame of axes oriented in the directions of the eigenvectors of \mathbf{U} . The equation of the ellipsoid along the original axes in \mathcal{J} -space is, of course,

$$\mathbf{t}^\top \mathbf{R} \mathbf{E} \boldsymbol{\Sigma}^{-2} \mathbf{E}^\top \mathbf{R}^\top \mathbf{t} = 1. \quad (10.13)$$

In summary the unit ball in joint space is mapped by the Jacobian-inverse \mathbf{J}^{-1} into an ellipsoid whose semi-axes are the singular values of \mathbf{J} . That is, \mathbf{J} *distorts* the unit ball in the joint-rate space into an ellipsoid in the EE-twist space. Hence, a measure of the quality of motion and force transmission of the robotic architecture from the joints to the EE is given by the above distortion; *the smaller the distortion, the higher the quality of the transmission*.

A measure of the Jacobian-induced distortion can thus be defined as the ratio of the largest σ_M to the smallest σ_m singular values of \mathbf{J} , which is nothing but the *condition number* κ_2 of \mathbf{J} based on the matrix 2-norm [10.31], i. e.,

$$\kappa_2 = \frac{\sigma_M}{\sigma_m}. \quad (10.14)$$

Actually, (10.14) is only one possibility of computing the condition number of \mathbf{J} , or of any $m \times n$ matrix for that matter, and certainly not the most economical. Notice that this definition requires knowledge of the singular values of the Jacobian. However, computing the singular values of a matrix is as computationally intensive as computing eigenvalues, with the added cost of the polar decomposition; the combined operation is slightly less expensive than the singular-value decomposition [10.32]. The most general definition of condition number, for $n \times n$ matrices, is [10.31]

$$\kappa(\mathbf{A}) = \|\mathbf{A}\| \|\mathbf{A}^{-1}\|. \quad (10.15)$$

As stated above, the expression (10.14) is obtained when the matrix 2-norm is adopted in (10.15). The matrix 2-norm is defined as

$$\|\mathbf{A}\|_2 \equiv \max_i \{ \sigma_i \}. \quad (10.16)$$

If, on the other hand, the *weighted matrix Frobenius norm* is adopted, which is defined as

$$\|\mathbf{A}\|_F \equiv \sqrt{\frac{1}{n} \text{tr}(\mathbf{A} \mathbf{A}^\top)} \equiv \sqrt{\frac{1}{n} \text{tr}(\mathbf{A}^\top \mathbf{A})} \quad (10.17)$$

then, apparently, the computation of the singular values can be obviated. When the weight $1/n$ is omitted in the above definition, the standard Frobenius norm is obtained. Notice, however, that the weighted Frobenius norm is more significant in engineering, for it does not depend on the number of rows and columns of the matrix at hand. The weighted Frobenius norm, in fact, yields the root-mean-square (rms) value of the set of singular values.

The *Frobenius condition number* κ_F of the Jacobian \mathbf{J} , based on the matrix Frobenius norm, is then

$$\begin{aligned} \kappa_F(\mathbf{J}) &= \frac{1}{n} \sqrt{\text{tr}(\mathbf{J} \mathbf{J}^\top)} \sqrt{\text{tr}[(\mathbf{J} \mathbf{J}^\top)^{-1}]} \\ &= \frac{1}{n} \sqrt{\text{tr}(\mathbf{J}^\top \mathbf{J})} \sqrt{\text{tr}[(\mathbf{J}^\top \mathbf{J})^{-1}]}. \end{aligned} \quad (10.18)$$

One more important difference between the two forms of computing the matrix condition number is worth pointing out: $\kappa_2(\cdot)$ is not an *analytic function* of its matrix argument, while $\kappa_F(\cdot)$ is. Hence, the condition number based on the Frobenius norm can be applied to great advantage in robot architecture design, as $\kappa_F(\cdot)$ is differentiable and lends itself to gradient-dependent optimization methods, which are much faster than direct methods based only on function evaluations. In robot control, which requires real-time computations, $\kappa_F(\cdot)$ also offers practical advantages, for its computation

obviates the knowledge of the singular values, only requiring a matrix inversion, which is a rather simple task.

The significance of the condition number in robot design and control can be best understood if we recall that the concept stems from numerical analysis in connection with the solution of the linear system of equations (10.1) for $\dot{\theta}$. Given that J is a function of both the architecture parameters and the posture variables θ , J is known only to within the error with which those quantities are known. Further, let the architecture parameters, namely, the constant values in the Denavit–Hartenberg parameter list, be stored in an array p . Both p and θ are known up to measurement errors δp and $\delta \theta$, respectively. Moreover, the twist t is input into the control software of the robot with an unavoidable error δt .

In solving (10.1) for $\dot{\theta}$ with floating-point arithmetic, the computed value is contaminated with a roundoff error $\delta \dot{\theta}$. The *relative error* in the computed $\dot{\theta}$ is related to the relative errors in the architecture parameters and posture variables by the relation [10.31]

$$\frac{\|\delta \dot{\theta}\|}{\|\dot{\theta}\|} \leq \kappa(J) \left(\frac{\|\delta p\|}{\|p\|} + \frac{\|\delta \theta\|}{\|\theta\|} + \frac{\|\delta t\|}{\|t\|} \right), \quad (10.19)$$

where p and θ represent the (unknown) actual values of these vectors and t the nominal value of the twist.

Nevertheless, the foregoing discussion applies to tasks involving either positioning or orientation, but not both. Most frequently, robotic tasks involve both positioning and orientation, which leads to Jacobian matrices with entries bearing disparate physical units, the consequence being that the Jacobian singular values have disparate units as well. Indeed, singular values associated with positioning bear units of length, whereas those associated with orientation are dimensionless. As a consequence, it is impossible to either order all singular values from smallest to largest or to add them all.

To cope with the issue of disparate units in the Jacobian entries, and to allow for the computation of the Jacobian condition number, the concept of *characteristic length* has been proposed [10.11]. The characteristic length L has been defined as the length by which the entries of the Jacobian with units of length are to be divided to render the Jacobian condition number a minimum at an optimum posture. This definition, while sound, lacks an immediate geometric interpretation, which has made its adoption in the robotics community difficult. In order to provide for a geometric interpretation, the concept of *homogeneous space* was recently introduced [10.33]. Using this concept, the robot architecture is designed in a *dimensionless space*, with points whose coordinates are dimensionless real numbers. In this way, the

six Plücker coordinates [10.34] of lines are all dimensionless, and hence the columns of the robot Jacobian matrix, comprising the Plücker coordinates of the revolute axes, are dimensionless as well. As a consequence, the Jacobian singular values are all dimensionless, and the Jacobian condition number can be defined. Once the robotic architecture has been designed for minimum condition number, under geometric constraints on link-length ratios and angles between neighboring revolute axes, for example, the maximum reach of the robot can be calculated. The maximum reach r will thus be a dimensionless quantity. When this quantity is compared with the prescribed maximum reach R , with units of length, the characteristic length is computed as the ratio $L = R/r$.

10.3.2 Dynamics-Based Local Performance Evaluation

Since motions are caused by forces and torques acting on rigid bodies, it seems reasonable to formulate performance indices that take into account the inertial properties of the mechanism. Asada [10.35] defines the *generalized-inertia ellipsoid* (GIE) as the ellipsoid defined by the product $G = J^{-T} M J^{-1}$, where M denotes the inertia matrix of the manipulator. This ellipsoid is that with semiaxes given by the singular values of the foregoing product. Yoshikawa [10.36], in turn, defines a corresponding dynamic manipulability measure as $\det[J M^{-1} (J M^{-1})^T]$. Physically these concepts represent two distinct phenomena. Suppose the robot is viewed as an input–output device which, given a joint torque, produces an acceleration at the end-effector. Yoshikawa’s index measures the uniformity of this torque–acceleration gain, whereas Asada’s GIE characterizes the inverse of this gain. If a human operator were holding the end-effector and attempting to move it about, the GIE would measure the resistance of the robot to this end-effector motion.

Other measures that attempt to capture robot performance as a function of the dynamics can be cited: Voglewede and Ebert–Uphoff [10.37] propose performance indices based on joint stiffness and link inertia, with the aim of establishing a distance to singularity for any robot posture, while Bowling and Khatib [10.38] propose a general framework for capturing the dynamic capability of a general robot manipulator that includes the velocity and acceleration characteristics of the end-effector, taking into account factors such as torque and the velocity limits of the actuators.

10.3.3 Global Dexterity Measures

The above measures are local in the sense that they characterize the dexterity of a robot at a given posture. Local measures are useful for applications ranging from redundancy resolution to workpiece positioning, but for design applications a global measure may be more desirable. One straightforward way of extending local measures to global ones is to integrate them over the allowable joint space. In [10.39] *Gosselin and Angeles* integrate the Jacobian condition number over the workspace to define a global measure, thereby producing a *global conditioning index*. For the simpler cases of planar positioning and spherical manipulators, the global conditioning index was found to coincide with its local counterpart.

10.3.4 Closed-Chain Dexterity Indices

The formulation of dexterity for closed chains presents a number of subtleties. The first obvious difference is that the joint configuration space of a closed chain will no longer be a flat space; in the general case it will be a curved multidimensional surface embedded in a higher-dimensional (typically flat) space. Also, dual to the open chain case, the forward kinematics for closed chains is generally more difficult to solve than the inverse kinematics, with the possibility of multiple solutions. Another important difference is that only a subset of the joints may be actuated, and that the number of actuated joints may even exceed the mechanism's kinematic degrees of freedom.

Several coordinate-based formulations for closed-chain dexterity have been proposed for specific mechanisms [10.40] and for cooperating robot systems whose joints are all assumed to be actuated [10.41, 42], at least some of which have led to apparently paradoxical results [10.43, 44]. Because of the nonlinear characteristics unique to closed-chain mechanisms discussed above, particular care must be exercised in formulating coordinate-based dexterity measures.

Another recent line of work has attempted a coordinate-invariant differential-geometric formulation of

dexterity for closed chains. In this framework the joint and end-effector configuration spaces are regarded as Riemannian manifolds with an appropriate choice of Riemannian metric, with the choice of joint space metric reflecting the characteristics of the joint actuators. The previous notions of ellipsoid developed for serial chains can be extended to general closed chains, containing both passive and active joints, and possibly redundantly actuated [10.45, 46].

10.3.5 Alternative Dexterity-Like Measures

The various definitions of dexterity discussed above all refer to the same qualitative feature of the ability of a robot to move and apply forces in arbitrary directions. A different viewpoint is taken in the work of *Liégeois* [10.47] and *Klein and Huang* [10.48], where dexterity is quantified in terms of joint-range availability. The driving motivation here lies in that most robots have joint limits; therefore, one should minimize the possibility of a joint reaching a stop.

Hollerbach [10.49] takes an alternative approach to designing a redundant 7R manipulator, by considering: (a) elimination of internal singularities, (b) ability to avoid obstacles in the workspace; (c) the solvability of the kinematic equations, and (d) mechanical constructability. Based on these four criteria, he derived a particular 7R design with the morphology of the human arm, i. e., composed of two spherical joints defining the shoulder and the wrist plus an intermediate revolute playing the role of the elbow. Now, while a redundant architecture should remain fully capable of performing six-DOF tasks upon locking one joint, the architecture reported in this reference may end up by losing this capability if the elbow joint is locked.

From a control perspective, *Spong* [10.50] shows that, if the inertia matrix of a manipulator has a vanishing Riemannian curvature, there exists a set of coordinates in which the equations of motion assume a particularly simple form. The curvature of the inertia matrix also reflects the sensitivity of the dynamics to certain robot parameters. Minimizing the curvature, therefore, is another possible criterion for robot design.

10.4 Other Performance Indices

10.4.1 Acceleration Radius

Another measure that attempts to capture the dynamic capabilities of a robotic manipulator is the *acceler-*

ation radius. Initially proposed by *Graettinger and Krogh* in [10.51], the acceleration radius measures the minimum acceleration capability of the end-effector, in arbitrary directions, for the given torque bounds on the

actuators. Specifically, given the dynamics equations for a serial chain in the form

$$\tau = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} \quad (10.20)$$

in which M is the robot *mass matrix* – also known as the *inertia matrix* – in *joint space*, and $C(\theta, \dot{\theta})$ is the matrix mapping the joint-rate vector to the vector of Coriolis and centrifugal forces in the same space. Moreover, the actuators are assumed to have joint torque limits of the form

$$\tau_{\min} \leq \tau \leq \tau_{\max}, \quad (10.21)$$

where the lower and upper limits $\tau_{\min}, \tau_{\max} \in \mathbb{R}^n$ are constant or functions of the manipulator posture θ . The end-effector *twist rate*, denoted by \dot{i} , can be expressed as

$$\dot{i} = J(\theta)\ddot{\theta} + J(\theta, \dot{\theta})\dot{\theta}, \quad (10.22)$$

where $J(\theta, \dot{\theta})$ is the Jacobian time-derivative. The Jacobian J was introduced in (10.1).

Under the assumption that $J(\theta)$ is nonsingular, one can write

$$\ddot{\theta} = J(\theta)^{-1}\dot{i} - J(\theta)^{-1}J(\theta, \dot{\theta})\dot{\theta}. \quad (10.23)$$

Substituting the above expression into the dynamic equations (10.20) leads to

$$\tau(\theta, t, \dot{i}) = M'(\theta)\dot{i} + C'(\theta, t), \quad (10.24)$$

where

$$M'(\theta) = M(\theta)J(\theta)^{-1} \\ C'(\theta, t) = [C(\theta, t) - M(\theta)J(\theta)^{-1}J(\theta, \dot{\theta})]J^{-1}(\theta).$$

For a given state $(\theta, \dot{\theta})$, the linear torque bounds (10.21) now define a polytope in the twist-rate space. *Graettinger* and *Krogh* [10.51] define the acceleration radius to be the largest sphere centered at the origin that is contained in this polytope; the radius reflects the minimal guaranteed EE acceleration in arbitrary directions. This concept is applied to measure the EE acceleration capability of a manipulator, as well as to determine the actuator sizes for achieving a desired acceleration radius. *Bowling* and *Khatib* [10.38] generalize this concept to capture both force and acceleration capabilities of the end-effector, with a view to quantifying the worst-case dynamic performance capability of a manipulator.

10.4.2 Elastostatic Performance

Elastostatic performance refers to the robotic system's response to the applied load – force and moment, i. e.,

wrench – under static equilibrium. This response may be measured in terms of the *stiffness* of the manipulator, which determines the translation and angular deflections when the EE is subjected to an applied wrench.

Robot deflections have two sources: link and joint deflection. In the presence of long links, as in the space robot *Canadarm2*, link compliance is a major source of deflection. However, in the majority of today's serial robots, the main source of deflection occurs at the joints.

In this chapter, we assume that the manipulator links are rigid, and model the joints as linearly elastic torsional springs. The more complex problem of link flexibility is studied in detail in Chap. 13. That is, for the elastostatic model we base our analysis on the assumption that, under a positioning task, the joints are locked at a certain posture θ_0 , while the EE is subject to a perturbation wrench Δw that is balanced by an elastic joint torque $\Delta \tau$. Under these conditions, $\Delta \theta$ and $\Delta \tau$ obey the well-known linear relation

$$K\Delta\theta = \Delta\tau \quad (10.25)$$

in which K is the *stiffness matrix* in joint space at the given posture. Moreover, the matrix K is diagonal, with its entries equal to the torsional stiffnesses of the corresponding joints, K is therefore *posture independent*, i. e., constant throughout the whole robot workspace. Moreover, since all joints exhibit a finite, nonzero stiffness, K is invertible, its inverse C being termed the *compliance matrix*. We can thus express the inverse relation of (10.25) as

$$\Delta\theta = C\Delta\tau. \quad (10.26)$$

It should be apparent that $\Delta\theta$ and $\Delta\tau$ have an *incremental* nature, for both are measured from the equilibrium posture, at which $\Delta\tau$ and $\Delta\theta$ vanish.

On the issue of stiffness matrix, *Griffis* and *Duffy* [10.52] proposed a mapping from an incremental rigid-body displacement Δx into an incremental wrench Δw that turned out to be nonsymmetric. The concept behind this mapping was formalized by *Howard* et al. [10.53] by means of Lie algebra. However, in the foregoing papers, Δx and Δw turn out to be incompatible in the sense that their reciprocal product does not yield incremental work – the point at which Δx is defined is distinct from that at which Δw is applied. For this reason, the array representation of the same mapping cannot be, properly speaking, termed a stiffness matrix.

For a constant magnitude of $\Delta\tau$, the deflection attains its maximum value in the direction of the eigenvector associated with the maximum eigenvalue of C

or, equivalently, with the minimum eigenvalue of \mathbf{K} , denoted by κ_{\min} . In terms of elastostatic performance, we aim to (a) make the maximum deflection a minimum, i. e., we want to maximize κ_{\min} , and (b) make the magnitude of the deflection $\|\Delta\theta\|$ as insensitive as possible to changes in the direction of the applied load $\Delta\tau$. This can be done by rendering κ_{\min} as close as possible to κ_{\max} . The first aim is associated with the stiffness constants, i. e., the higher the constants the lower the deflections. The later, however, is associated with the concept of isotropy, the ideal case being when all the eigenvalues of \mathbf{K} are identical, which means that \mathbf{K} is isotropic. Due to the *pyramidal effect* of serial robots, in which downstream motors carry their upstream counterparts, the joint stiffness is bound to be largest for the proximal joints. Hence an isotropic stiffness matrix is impossible for serial robots.

Notice that (10.25) and (10.26) may also be formulated in the task space, i. e.,

$$\mathbf{K}_C \Delta \mathbf{x} = \Delta \mathbf{w}, \quad (10.27)$$

where $\Delta \mathbf{x} \equiv \mathbf{t} \Delta t$, with Δt denoting a *small* time interval producing a correspondingly *small* change $\Delta \mathbf{x}$ in the pose of the EE. That is,

$$\Delta \mathbf{x} = \mathbf{J} \dot{\theta} \Delta t = \mathbf{J} \Delta \theta \quad (10.28)$$

which is a linear transformation of the joint-vector increment into the pose-increment vector. We show next that *the stiffness matrix is not frame invariant*. This means that, under the linear transformation from joint to Cartesian coordinates, *the stiffness matrix does not obey a similarity transformation*. For quick reference, we recall below the definition of similarity transformation: if $\mathbf{y} = \mathbf{L}\mathbf{x}$ is a linear transformation of \mathbb{R}^n into itself, and we introduce a change of vector basis, $\mathbf{x}' = \mathbf{A}\mathbf{x}$, $\mathbf{y}' = \mathbf{A}\mathbf{y}$, then \mathbf{L} changes to \mathbf{L}' , which is given by

$$\mathbf{L}' = \mathbf{A} \mathbf{L} \mathbf{A}^{-1}. \quad (10.29)$$

The above transformation of any vector of \mathbb{R}^n into another one of the same space, and of a matrix \mathbf{L} into \mathbf{L}' , as given by (10.29), is termed a *similarity transformation*. Notice that \mathbf{A} is guaranteed to be invertible, as it represents a change of coordinate frame.

Now, under the change of coordinates given by (10.28), (10.27) leads to

$$\mathbf{K}_C \mathbf{J} \Delta \theta = \mathbf{J}^{-\top} \Delta \tau, \quad (10.30)$$

where we have used the kinetostatic relation [10.11]

$$\mathbf{J}^\top \Delta \mathbf{w} = \Delta \tau.$$

The exponent $-\top$ denotes the transpose of the inverse or, equivalently, the inverse of the transpose. Upon multiplication of both sides of (10.30) by \mathbf{J}^\top from the left, we end up with

$$\mathbf{J}^\top \mathbf{K}_C \mathbf{J} \Delta \theta = \Delta \tau. \quad (10.31)$$

If we now compare (10.25) with (10.31), we can readily derive the relations between the stiffness matrix \mathbf{K} in joint space and the stiffness matrix \mathbf{K}_C in Cartesian space:

$$\mathbf{K} = \mathbf{J}^\top \mathbf{K}_C \mathbf{J} \quad \text{and} \quad \mathbf{K}_C = \mathbf{J}^{-\top} \mathbf{K} \mathbf{J}^{-1} \quad (10.32)$$

which apparently is not a similarity transformation between \mathbf{K} and \mathbf{K}_C . What this means is that the two matrices do not share the same set of eigenvalues and their eigenvectors are not related by the linear relation (10.28). As a matter of fact, if the robot is revolute-coupled, the entries of its stiffness matrix \mathbf{K} all have units of Nm, i. e., of *torsional stiffness*, while the entries of \mathbf{K}_C have disparate units. To show this, the Jacobian matrix, the inverse Jacobian and the two stiffness matrices are partitioned correspondingly into four 3×3 blocks, i. e., as

$$\mathbf{J} = \begin{pmatrix} \mathbf{J}_{11} & \mathbf{J}_{12} \\ \mathbf{J}_{21} & \mathbf{J}_{22} \end{pmatrix}, \quad \mathbf{J}^{-1} = \begin{pmatrix} \mathbf{J}'_{11} & \mathbf{J}'_{12} \\ \mathbf{J}'_{21} & \mathbf{J}'_{22} \end{pmatrix},$$

$$\mathbf{K} = \begin{pmatrix} \mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{12}^\top & \mathbf{K}_{22} \end{pmatrix}, \quad \mathbf{K}_C = \begin{pmatrix} \mathbf{K}_{C11} & \mathbf{K}_{C12} \\ \mathbf{K}_{C12}^\top & \mathbf{K}_{C22} \end{pmatrix}.$$

Furthermore, in light of the definition of the twist, (10.2), the two upper blocks of \mathbf{J} are dimensionless, while its two lower blocks have units of length [10.11]. As a consequence, the two left blocks of \mathbf{J}^{-1} are dimensionless, while its two right blocks have units of inverse length. Now, the blocks of \mathbf{K}_C are computed from the corresponding relation in (10.32), thereby obtaining

$$\begin{aligned} \mathbf{K}_{C11} &= \mathbf{J}_{11}^\top (\mathbf{K}_{11} \mathbf{J}'_{11} + \mathbf{K}_{12} \mathbf{J}'_{21}) \\ &\quad + \mathbf{J}_{21}^\top (\mathbf{K}_{12}^\top \mathbf{J}'_{11} + \mathbf{K}_{22} \mathbf{J}'_{21}) \\ \mathbf{K}_{C12} &= \mathbf{J}_{11}^\top (\mathbf{K}_{11} \mathbf{J}'_{12} + \mathbf{K}_{12} \mathbf{J}'_{22}) \\ &\quad + \mathbf{J}_{21}^\top (\mathbf{K}_{12}^\top \mathbf{J}'_{12} + \mathbf{K}_{22} \mathbf{J}'_{22}) \\ \mathbf{K}_{C21} &= \mathbf{K}_{C12}^\top \\ \mathbf{K}_{C22} &= \mathbf{J}_{12}^\top (\mathbf{K}_{11} \mathbf{J}'_{12} + \mathbf{K}_{12} \mathbf{J}'_{22}) \\ &\quad + \mathbf{J}_{22}^\top (\mathbf{K}_{12}^\top \mathbf{J}'_{12} + \mathbf{K}_{22} \mathbf{J}'_{22}). \end{aligned}$$

It is now apparent that: K_{C11} has entries with units of Nm, i. e., of torsional stiffness; K_{C12} and K_{C21} have entries with units of N; and K_{C22} has entries with units of N/m, i. e., of translational stiffness.

The outcome of the foregoing discussion is that a norm for K is possible, but not one for K_C , unless of course a characteristic length is introduced to render all the entries of K_C dimensionally homogeneous. A norm of a matrix is useful as it indicates *how large* the entries of the matrix are. We would like to characterize how stiff a robot is in both joint and Cartesian spaces. In joint space we could adopt any norm, but notice that the 2-norm, introduced in (10.3), would be misleading, as this norm would assign the value of the strongest joint to the overall robot stiffness. A more suitable norm would be the weighted Frobenius norm, introduced in (10.17), which would assign the rms value of the joint stiffnesses to the overall robot stiffness.

To design a robot optimally, we would therefore aim to maximize the Frobenius norm of its stiffness matrix in joint space, while observing constraints on the robot weight, as stiffer joints lead to heavier joints if the same material is used for all joints.

10.4.3 Elastodynamic Performance

For a general design problem, not only the kinetostatic and elastostatic performances, but also the elastodynamic performance have to be considered. In this regard, we introduce the assumptions of Sect. 10.4.2, with the added condition that inertia forces due to the link masses and moments of inertia are now taken into consideration.

The linearized model of a serial robot at the posture given by θ_0 , if we neglect damping, is

$$M\Delta\ddot{\theta} + K\Delta\theta = \Delta\tau \quad (10.33)$$

in which M is the $n \times n$ positive-definite mass matrix introduced in Sect. 10.4.1, while K was defined in Sect. 10.4.2 as the $n \times n$ positive-definite stiffness matrix in joint space. Both K and M were defined in joint-space coordinates, $\Delta\theta$, representing the vector of joint-variable elastic displacements, as in Sect. 10.4.2. These displacements are produced when, as the joints are all locked at a value θ_0 and thereby become ideal linear springs, the robot is subject to a perturbation $\Delta\tau$, to nonzero initial conditions, or to a combination of both.

Under *free vibration*, i. e., under a motion of the system (10.33) caused by nonzero initial conditions and a zero excitation $\Delta\tau$, the foregoing equation can be

solved for $\Delta\ddot{\theta}$:

$$\Delta\ddot{\theta} = -D\Delta\theta, \quad D \equiv M^{-1}K \quad (10.34)$$

in which the matrix D is known as the *dynamic matrix*. This matrix determines the behavior of the system under consideration, as its eigenvalues are the *natural frequencies* of the system and its eigenvectors the *modal vectors*. Let $\{\omega_i\}_1^n$ and $\{f_i\}_1^n$ denote the sets of eigenvalues and the corresponding eigenvectors of D , respectively. Under the initial conditions $[\Delta\theta(0), \Delta\dot{\theta}(0)]^\top$, in which $\Delta\theta(0)$ is proportional to the i -th eigenvector of D and $\Delta\dot{\theta}(0) = \mathbf{0}$, the ensuing motion is of the form $\Delta\theta(t) = \Delta\theta(0) \cos \omega_i t$ [10.54].

Furthermore, under the change of variables given by (10.28), the model (10.33) changes to

$$MJ^{-1}\Delta\ddot{x} + KJ^{-1}\Delta x = J^\top \Delta w.$$

If we now multiply both sides of the above equation by $J^{-\top}$, we obtain the elastodynamic model (10.33) in Cartesian coordinates, namely

$$J^{-\top}MJ^{-1}\Delta\ddot{x} + J^{-\top}KJ^{-1}\Delta x = \Delta w$$

in which the first matrix coefficient is the mass matrix M_C in Cartesian coordinates, and the second is identified as K_C , which was introduced in (10.32), i. e.,

$$M_C \equiv J^{-\top}MJ^{-1}. \quad (10.35)$$

The elastodynamic model in Cartesian coordinates thus takes the form

$$M_C\Delta\ddot{x} + K_C\Delta x = \Delta w. \quad (10.36)$$

Again, by virtue of the transformation (10.35), it is apparent that the mass matrix, like its stiffness counterpart, is not invariant under a change of coordinates. Moreover, in a revolute-coupled robot, all the entries of M have units of kg m^2 ; however, the entries of M_C have disparate units. An analysis similar to that conducted in Sect. 10.4.2 for the stiffness matrix in Cartesian space reveals that, if M_C is partitioned into four 3×3 blocks, then its upper-left block has units of moment of inertia, its lower-right block has units of mass, while its off-diagonal blocks have units of kg m .

Correspondingly, the dynamic matrix in Cartesian coordinates becomes

$$D_C = M_C^{-1}K_C. \quad (10.37)$$

It is now a simple matter to prove that *the dynamic matrix is invariant under a change of coordinates*. Indeed,

if we substitute transformations (10.32) and (10.35) into (10.37), we obtain

$$\mathbf{D}_C = \mathbf{J}\mathbf{M}^{-1}\mathbf{J}^\top \mathbf{J}^{-\top} \mathbf{K}\mathbf{J}^{-1} = \mathbf{J}\mathbf{M}^{-1}\mathbf{K}\mathbf{J}^{-1},$$

in which the dynamic matrix \mathbf{D} in joint coordinates can be readily identified in the final expression, and hence,

$$\mathbf{D}_C = \mathbf{J}\mathbf{D}\mathbf{J}^{-1}, \quad (10.38)$$

which shows that \mathbf{D}_C is indeed a similarity transformation of \mathbf{D} . As a consequence, the dynamic matrix is indeed invariant under a change of coordinates, the two matrices thus sharing the same sets of eigenvalues, their eigenvectors being related by the same similarity transformation. That is, if the modal vectors in joint space – the eigenvectors of \mathbf{D} – are denoted by $\{\mathbf{f}_i\}_1^n$ and the modal vectors in Cartesian space are denoted by $\{\mathbf{g}_i\}_1^n$, then these two sets are related by

$$\mathbf{g}_i = \mathbf{J}\mathbf{f}_i, \quad i = 1, \dots, n. \quad (10.39)$$

Therefore, the natural frequencies of the elastodynamic model are the same, regardless of the space in which they are calculated, while their natural modes of vibration change under a similarity transformation.

Under an excitation of the form $\Delta\tau = \theta_0 \cos \omega t$ and zero initial conditions, the system is known to respond

with a harmonic motion of frequency ω and magnitude that depends on both ω and the system *frequency spectrum* $\{\omega_i\}_1^n$ [10.54]. When ω equals one of the natural frequencies of the system, the response magnitude grows unbounded, a phenomenon known as *resonance*. For this reason, when designing a robot, it is imperative that its frequency spectrum does not involve any of the expected operation frequencies, which can be achieved by designing the robot with stiffness and mass matrices that yield a frequency spectrum outside of the frequency range of the operation conditions.

This design task is not straightforward. Indeed, while the stiffness matrix in joint space is constant, the mass matrix is posture dependent, i.e., $\mathbf{M} = \mathbf{M}(\boldsymbol{\theta})$. Because of this feature, the elastodynamic design of a robot is bound to be iterative: the design can be conducted for a *straw-man task*, i.e., a typical task in the target applications, thus defining a set of postures that lead in turn to a set of mass-matrix values. Then, the frequency spectrum for all these postures can be designed to lie above the frequency range of the straw-man task. Since the robot will eventually be commanded to execute other tasks than the straw-man task, simulation of alternative tasks is required to ensure that the design is safe from a resonance viewpoint.

References

- 10.1 O. Bottema, B. Roth: *Theoretical Kinematics* (North-Holland, Amsterdam 1979), Also available by Dover Publishing, New York 1990
- 10.2 J. Angeles: The qualitative synthesis of parallel manipulators, *ASME J. Mech. Des.* **126**(4), 617–624 (2004)
- 10.3 J. Denavit, R.S. Hartenberg: A kinematic notation for lower-pair mechanisms based on matrices, *ASME J. Appl. Mech.* **77**, 215–221 (1955)
- 10.4 G. Pahl, W. Beitz: *Engineering Design. A Systematic Approach*, 3rd edn. (Springer, London 2007), Translated from the original Sixth Edition in German
- 10.5 M. Petterson, J. Andersson, P. Krus, X. Feng, D. Wappling: Industrial Robot Design Optimization in the Conceptual Design Phase, *Proc. Mechatron. Robot.*, Vol. 2, ed. by P. Drews (APS-European Centre for Mechatronics, Aachen 2004) pp.125–130
- 10.6 Koning & Hartman, Amsterdam, The Netherlands, http://www.konighartman.com/nl/producten/aandrijven_en_besturen/robots/ir_rp_ah/ (November 23, 2007)
- 10.7 R. Clavel: Device for the movement and positioning of an element in space, Patent 4976582 (1990)
- 10.8 R. Vijaykumar, K.J. Waldron, M.J. Tsai: Geometric optimization of serial chain manipulator structures for working volume and dexterity, *Int. J. Robot. Res.* **5**(2), 91–103 (1986)
- 10.9 J.P. Merlet: *Parallel Robots* (Springer, Dordrecht 2006)
- 10.10 K. Hoffman: *Analysis in Euclidean Space* (Prentice-Hall, Englewood Cliffs 1975)
- 10.11 J. Angeles: *Fundamentals of Robotic Mechanical Systems*, 3rd edn. (Springer, New York 2007)
- 10.12 L. Burmester: *Lehrbuch der Kinematik* (Arthur Felix, Leipzig 1886), in German
- 10.13 J.M. McCarthy: *Geometric Design of Linkages* (Springer, New York 2000)
- 10.14 H. Li: Ein Verfahren zur vollständigen Lösung der Rückwärtstransformation für Industrieroboter mit allgemeiner Geometrie. Ph.D. Thesis (Universität-Gesamthochschule Duisburg, Duisburg 1990)
- 10.15 M. Raghavan, B. Roth: Kinematic analysis of the 6R manipulator of general geometry, *Proc. 5th Int. Symp. Robot. Res.*, ed. by H. Miura, S. Arimoto (MIT Press, Cambridge 1990)
- 10.16 J. Angeles: The degree of freedom of parallel robots: a group-theoretic approach, *Proc. IEEE Int. Conf. Robot. Autom.* (Barcelona 2005) pp.1017–1024
- 10.17 C.C. Lee, J.M. Hervé: Translational parallel manipulators with doubly planar limbs, *Mechanism Machine Theory* **41**, 433–455 (2006)

- 10.18 B. Roth: Performance evaluation of manipulators from a kinematic viewpoint, National Bureau of Standards – NBS SP **495**, 39–61 (1976)
- 10.19 A. Kumar, K.J. Waldron: The workspaces of a mechanical manipulator, ASME J. Mech. Des. **103**, 665–672 (1981)
- 10.20 D.C.H. Yang, T.W. Lee: On the workspace of mechanical manipulators, ASME J. Mech. Trans. Autom. Des. **105**, 62–69 (1983)
- 10.21 Y.C. Tsai, A.H. Soni: An algorithm for the workspace of a general n-R robot, ASME J. Mech. Trans. Autom. Des. **105**, 52–57 (1985)
- 10.22 K.C. Gupta, B. Roth: Design considerations for manipulator workspace, ASME J. Mech. Des. **104**, 704–711 (1982)
- 10.23 K.C. Gupta: On the nature of robot workspace, Int. J. Robot. Res. **5**(2), 112–121 (1986)
- 10.24 F. Freudenstein, E. Primrose: On the analysis and synthesis of the workspace of a three-link, turning-pair connected robot arm, ASME J. Mech. Trans. Autom. Des. **106**, 365–370 (1984)
- 10.25 C.C. Lin, F. Freudenstein: Optimization of the workspace of a three-link turning-pair connected robot arm, Int. J. Robot. Res. **5**(2), 91–103 (1986)
- 10.26 T. Yoshikawa: Manipulability of robotic mechanisms, Int. J. Robot. Res. **4**(2), 3–9 (1985)
- 10.27 J. Loncaric: Geometric Analysis of Compliant Mechanisms in Robotics. Ph.D. Thesis (Harvard University, Harvard 1985)
- 10.28 B. Paden, S. Sastry: Optimal kinematic design of 6R manipulators, Int. J. Robot. Res. **7**(2), 43–61 (1988)
- 10.29 J.K. Salisbury, J.J. Craig: Articulated hands: force control and kinematic issues, Int. J. Robot. Res. **1**(1), 4–17 (1982)
- 10.30 G. Strang: *Linear Algebra and Its Applications*, 3rd edn. (Harcourt Brace Jovanovich College Publishers, New York 1988)
- 10.31 G.H. Golub, C.F. Van Loan: *Matrix Computations* (The Johns Hopkins Univ. Press, Baltimore 1989)
- 10.32 A. Dubrulle: An optimum iteration for the matrix polar decomposition, Electron. Trans. Numer. Anal. **8**, 21–25 (1999)
- 10.33 W.A. Khan, J. Angeles: The Kinetostatic Optimization of Robotic Manipulators: The Inverse and the Direct Problems, ASME J. Mech. Des. **128**, 168–178 (2006)
- 10.34 H. Pottmann, J. Wallner: *Computational Line Geometry* (Springer, Berlin, Heidelberg, New York 2001)
- 10.35 H. Asada: A geometrical representation of manipulator dynamics and its application to arm design, Trans. ASME J. Dyn. Sys. Meas. Contr. **105**(3), 131–135 (1983)
- 10.36 T. Yoshikawa: Dynamic manipulability of robot manipulators, Proc. IEEE Int. Conf. Robot. Autom. (1985) pp. 1033–1038
- 10.37 P.A. Voglewede, I. Ebert-Uphoff: Measuring closeness to singularities for parallel manipulators, Proc. IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp. 4539–4544
- 10.38 A. Bowling, O. Khatib: The dynamic capability equations: a new tool for analyzing robotic manipulator performance, IEEE Trans. Robot. **21**(1), 115–123 (2005)
- 10.39 C.M. Gosselin, J. Angeles: A new performance index for the kinematic optimization of robotic manipulators, Proc. 20th ASME Mech. Conf. (Kissimmee 1988) pp. 441–447
- 10.40 C. Gosselin, J. Angeles: The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator, ASME J. Mech. Trans. Autom. Des. **110**, 35–41 (1988)
- 10.41 A. Bicchi, C. Melchiorri, D. Balluchi: On the mobility and manipulability of general multiple limb robots, IEEE Trans. Robot. Autom. **11**(2), 232–235 (1995)
- 10.42 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Global task space manipulability ellipsoids for multiple-arm systems, IEEE Trans. Robot. Autom. **7**, 678–685 (1991)
- 10.43 C. Melchiorri: Comments on Global task space manipulability ellipsoids for multiple-arm systems and further considerations, IEEE Trans. Robot. Autom. **9**, 232–235 (1993)
- 10.44 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Reply to comments on Global task space manipulability ellipsoids for multiple-arm systems' and further considerations, IEEE Trans. Robot. Autom. **9**, 235–236 (1993)
- 10.45 F.C. Park: Optimal robot design and differential geometry, ASME Special 50th Anniv. Design Issue **117**(B), 87–92 (1995)
- 10.46 F.C. Park, J. Kim: Manipulability of closed kinematic chains, ASME J. Mech. Des. **120**(4), 542–548 (1998)
- 10.47 A. Liégeois: Automatic supervisory control for the configuration and behavior of multibody mechanisms, IEEE Trans. Sys. Man. Cyber. **7**(12), 842–868 (1977)
- 10.48 C.A. Klein, C.H. Huang: Review of pseudo-inverse control for use with kinematically redundant manipulators, IEEE Trans. Sys. Man. Cyber. **13**(2), 245–250 (1983)
- 10.49 J.M. Hollerbach: Optimum kinematic design of a seven degree of freedom manipulator. In: *Robotics Research: The Second International Symposium*, ed. by H. Hanafusa, H. Inoue (MIT Press, Cambridge 1985)
- 10.50 M.W. Spong: Remarks on robot dynamics: canonical transformations and riemannian geometry, Proc. IEEE Int. Conf. Robot. Autom. (1992) pp. 454–472
- 10.51 T.J. Graettinger, B.H. Krogh: The acceleration radius: a global performance measure for robotic

manipulators, IEEE J. Robot. Autom. **4**(11), 60–69 (1988)

10.52 M. Griffis, J. Duffy: Global stiffness modeling of a class of simple compliant couplings, Mechanism Machine Theory **28**, 207–224 (1993)

10.53 S. Howard, M.J. Zefran Kumar: On the 6×6 cartesian stiffness matrix for three-dimensional motions, Mechanism and Machine Theory **33**, 389–408 (1998)

10.54 L. Meirovitch: *Fundamentals of vibrations* (McGraw-Hill, Boston–London 2001)

Kinematically Redundant Manipulators

Stefano Chiaverini, Giuseppe Oriolo, Ian D. Walker

This chapter focuses on redundancy resolution schemes, i. e., the techniques for exploiting the redundant degrees of freedom in the solution of the inverse kinematics problem. This is obviously an issue of major relevance for motion planning and control purposes.

In particular, task-oriented kinematics and the basic methods for its inversion at the velocity (first-order differential) level are first recalled, with a discussion of the main techniques for handling kinematic singularities. Next, different first-order methods to solve kinematic redundancy are arranged in two main categories, namely those based on the optimization of suitable performance criteria and those relying on the augmentation of the task space. Redundancy resolution methods at the acceleration (second-order differential) level are then considered in order to take into account dynamics issues, e.g., torque minimization. Conditions under which a cyclic task motion results in a cyclic joint motion are also discussed; this is a major issue, e.g., for industrial applications in which a redundant manipulator is used to execute a repetitive task. The special class of hyperredundant manipulators is analyzed in detail. Suggestions for further reading are given in a final section.

11.1	Overview	245
11.2	Task-Oriented Kinematics	247
	11.2.1 Task-Space Formulation.....	247
	11.2.2 Singularities	248
11.3	Inverse Differential Kinematics	250
	11.3.1 The General Solution	250
	11.3.2 Singularity Robustness.....	251
	11.3.3 Joint Trajectory Reconstruction	254
11.4	Redundancy Resolution via Optimization	255
	11.4.1 Performance Criteria.....	255
	11.4.2 Local Optimization	256
	11.4.3 Global Optimization	256
11.5	Redundancy Resolution via Task Augmentation	256
	11.5.1 Extended Jacobian	256
	11.5.2 Augmented Jacobian	257
	11.5.3 Algorithmic Singularities	258
	11.5.4 Task Priority.....	258
11.6	Second-Order Redundancy Resolution	259
11.7	Cyclicity	260
11.8	Hyperredundant Manipulators	261
	11.8.1 Rigid-Link Hyperredundant Designs	261
	11.8.2 Continuum Robot Designs	263
	11.8.3 Hyperredundant Manipulator Modeling	263
11.9	Conclusion and Further Reading	265
	References	265

A kinematically redundant manipulator possesses more joints than those strictly required to execute its task. This provides the robot with an increased level of dexterity that may be used to avoid singularities, joint limits, and

workspace obstacles, but also to minimize joint torque, energy or, in general, to optimize suitable performance indexes.

11.1 Overview

Kinematic redundancy occurs when a robotic manipulator has more degrees of freedom than those strictly

required to execute a given task. This means that, in principle, no manipulator is inherently redundant; rather,

there are certain tasks with respect to which it may become redundant. Since it is widely recognized that a general task consists of following an end-effector motion trajectory requiring six degrees of freedom, a robot arm with seven or more joints is considered as the typical example of inherently redundant manipulator. However, even robot arms with fewer degrees of freedom, like conventional six-joint industrial manipulators, may become kinematically redundant for specific tasks, such as simple end-effector positioning without constraints on the orientation.

The motivation for introducing kinematic redundancy in the mechanical structure of a manipulator goes beyond that for using redundancy in traditional engineering design, namely, increasing robustness to faults so as to improve reliability (e.g., redundant processors or sensors). In fact, endowing robotic manipulators with kinematic redundancy is mainly aimed at increasing dexterity.

The minimal-complexity approach which characterized early manipulator design had the objective of minimizing cost and maintenance, for example, this led to the development of selective compliance assembly robot arm (SCARA) robots for pick-and-place operations. However, giving a manipulator the minimum number of joints required to execute its task results in a serious limitation in real-world applications where, in addition to the singularity problem, joint limits or workspace obstacles are present. All of these give rise to forbidden regions in the joint space that must be avoided during operation, thus requiring a carefully structured (and static) workspace in which the motion of the manipulator can be planned in advance; this is the case of workcells in traditional industrial applications.

On the other hand, the presence of additional degrees of freedom besides those strictly required to execute the



Fig. 11.1 The 7-DOF DLR Lightweight Robot

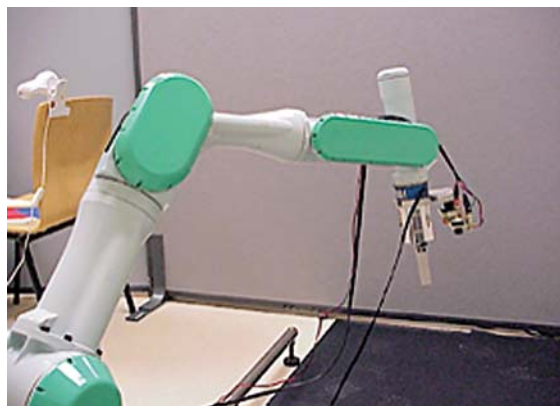


Fig. 11.2 The 7-DOF Mitsubishi PA-10 manipulator

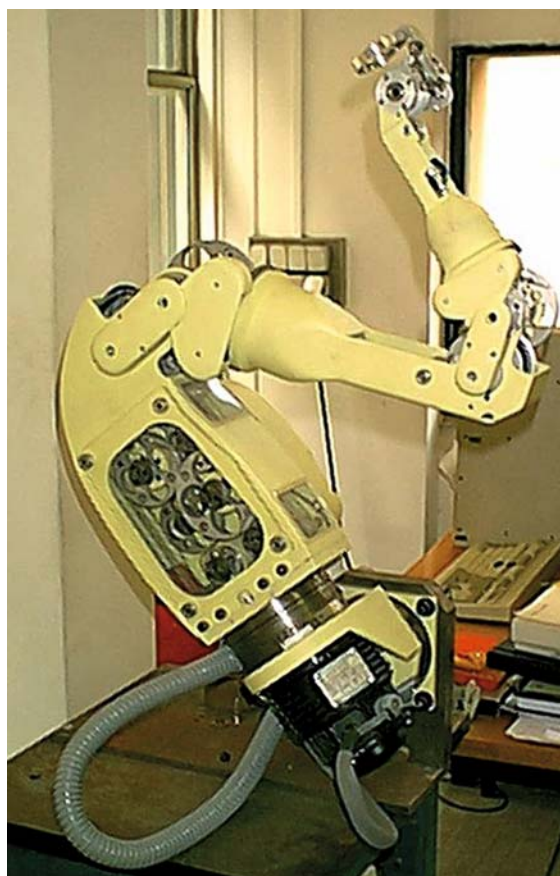


Fig. 11.3 The 8-DOF DEXTER by Scienza Machinale

task allows motions of the manipulator which do not displace the end effector (the so-called self-motions or internal motions); this implies that the same task at the

end-effector level can be executed in several ways at the joint level, giving the possibility of avoiding the forbidden regions and ultimately resulting in a more versatile mechanism. Such a feature is key to allowing operation in unstructured or dynamically varying environments that characterize advanced industrial applications and service robotics scenarios.

In practice, if properly managed, the increased dexterity characterizing kinematically redundant manipulators may allow to avoid singularities, joint limits, and workspace obstacles, but also to minimize torque/energy over a given task, ultimately meaning that the robotic manipulator can achieve a higher degree of autonomy.

The biological archetype of kinematically redundant manipulator is the human arm, which, not surprisingly, also inspires the terminology used to characterize the structure of serial-chain manipulators. In fact, the human arm has three degrees of freedom at the *shoulder*, one degree of freedom at the *elbow* and three degrees of freedom at the *wrist*. The available redundancy can be easily verified by locking one's wrist, e.g., on a table and moving the elbow while keeping the shoulder still. The kinematic arrangement of the human arm has been replicated in a number of

robots often termed as *human-arm-like* manipulators. This family of 7-DOF manipulators includes the DLR Lightweight Robot (Fig. 11.1) and the Mitsubishi PA-10 robot (Fig. 11.2). An example of an 8-DOF robot is the DEXTER by Scienza Macchine (Fig. 11.3). Manipulators with a larger number of joints are often called *hyperredundant* robots, and include many snake-like robots described in the literature.

The use of two or more robotic structures to execute a task (as in the case of cooperating manipulators or multifingered hands) also gives rise to kinematic redundancy. Redundant mechanisms also include vehicle-manipulator systems; in this case, however, the possible presence of nonholonomic constraints on the motion of the base must be properly taken into account in order to determine the actual degree of redundancy.

Although the realization of a kinematically redundant structure raises a number of issues from the point of view of mechanical design, this chapter focuses on redundancy resolution schemes, i.e., techniques for exploiting the redundant degrees of freedom in the solution of the inverse kinematics problem. This is an issue of major relevance for motion planning and control purposes.

11.2 Task-Oriented Kinematics

The relationship between the variables representing the configuration of an articulated manipulator and those describing an assigned task in the appropriate space can be established at the position, velocity or acceleration level. In particular, consideration of the first-order task kinematics brings up the task Jacobian matrix, which is a central object in redundancy resolution techniques.

11.2.1 Task-Space Formulation

A manipulator consists of a chain of rigid bodies articulated by joints. If q_i denotes the variable characterizing the relative displacement of body i with respect to body $i-1$, the vector $\mathbf{q} = (q_1 \dots q_N)^T$ uniquely describes the configuration of an N -joint serial-chain manipulator. Joint i may be either prismatic or revolute, in which case q_i measures the relative translation or rotation of the attached links, respectively.

While the manipulator is naturally described and actuated in the *joint space*, its operation is conveniently

specified in terms of the vector $\mathbf{t} = (t_1 \dots t_M)^T$, which typically defines the location of the manipulator end-effector in a suitably defined *task space*. In the general case, it is $M=6$ and \mathbf{t} is chosen so that its first three components represent the position of the end effector, while its last three components represent a minimal description of the end-effector orientation (such as Euler angles or the roll-pitch-yaw representation), i.e.,

$$\mathbf{t} = [p_x \ p_y \ p_z \ \alpha \ \beta \ \gamma]^T.$$

Typically, one has $N \geq M$, so that the joints can provide at least the number of degrees of freedom required for the end-effector task. If $N > M$ strictly, the manipulator is *kinematically redundant*.

The relationship between the joint-space coordinate vector \mathbf{q} and the task-space coordinate vector \mathbf{t} is expressed by the *direct kinematics* equation

$$\mathbf{t} = \mathbf{k}_t(\mathbf{q}), \quad (11.1)$$

where \mathbf{k}_t is a nonlinear vector function.

Task Jacobian and Geometric Jacobian

It is useful to consider the *first-order differential kinematics* [11.1]

$$\dot{\mathbf{i}} = \mathbf{J}_t(\mathbf{q}) \dot{\mathbf{q}}, \quad (11.2)$$

that can be obtained by differentiating (11.1) with respect to time. In (11.2), $\dot{\mathbf{i}}$ is the task-space velocity vector, $\dot{\mathbf{q}}$ is the joint-space velocity vector, and $\mathbf{J}_t(\mathbf{q}) = \partial \mathbf{k}_t / \partial \mathbf{q}$ is the $M \times N$ *task Jacobian* matrix (also called the *analytic Jacobian*).

Remarkably, the components of $\dot{\mathbf{i}}$ relative to the end-effector orientation express the rate of change of the parameters characterizing the adopted minimal representation; they are not the components of the angular velocity vector of the end-effector. Indeed, denoting by \mathbf{v}_N the 3×1 translational velocity vector and by $\boldsymbol{\omega}_N$ the 3×1 angular velocity vector of the end-effector, and defining the end-effector spatial velocity \mathbf{v}_N as

$$\mathbf{v}_N = \begin{pmatrix} \mathbf{v}_N \\ \boldsymbol{\omega}_N \end{pmatrix}, \quad (11.3)$$

the following relationship holds

$$\dot{\mathbf{i}} = \mathbf{T}(\mathbf{t}) \mathbf{v}_N, \quad (11.4)$$

where \mathbf{T} is an $M \times 6$ transformation matrix that is a function of \mathbf{t} only. In the case $M=6$, the transformation matrix \mathbf{T} takes the form

$$\mathbf{T} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{pmatrix}, \quad (11.5)$$

where \mathbf{I} and $\mathbf{0}$ are, respectively, the identity and null matrix of proper dimensions, and \mathbf{R} is a 3×3 matrix that specifically depends on the minimal representation used to describe the end-effector orientation.

For a given manipulator, the mapping

$$\mathbf{v}_N = \mathbf{J}(\mathbf{q}) \dot{\mathbf{q}} \quad (11.6)$$

relates a joint-space velocity to the corresponding end-effector velocity through the $6 \times N$ *geometric Jacobian* matrix \mathbf{J} . The geometric Jacobian matrix is of major concern in the kinematic analysis of a manipulator since it allows description of the motion capabilities of the end-effector (in terms of its free rigid-body spatial velocity) as a result of the velocity commands at the joints in the current configuration.

By comparing (11.2), (11.4), and (11.6), the relation between the geometric Jacobian and the task Jacobian can be found as

$$\mathbf{J}_t(\mathbf{q}) = \mathbf{T}(\mathbf{t}) \mathbf{J}(\mathbf{q}). \quad (11.7)$$

Second-Order Differential Kinematics

While the first-order differential kinematics (11.2) relates task-space to joint-space velocities, further differentiation with respect to time provides an analogous relationship between accelerations

$$\ddot{\mathbf{i}} = \mathbf{J}_t(\mathbf{q}) \ddot{\mathbf{q}} + \dot{\mathbf{J}}_t(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}}. \quad (11.8)$$

This equation is also known as the *second-order differential kinematics*.

11.2.2 Singularities

In this section the occurrence of singular configurations is considered to properly analyze their effect on inverse kinematics solutions.

Representation and Kinematic Singularities

A robot configuration \mathbf{q} is *singular* if the task Jacobian matrix \mathbf{J}_t is rank-deficient there. Considering the role of \mathbf{J}_t in (11.2) and (11.8), it is easy to realize that at a singular configuration it is impossible to generate end-effector task velocities or accelerations in certain directions. Further insight can be gained by looking at (11.7), which indicates that a singularity may be due to a loss of rank of the transformation matrix \mathbf{T} and/or of the geometric Jacobian matrix \mathbf{J} .

Rank deficiencies of \mathbf{T} are only related to the mathematical relationship established by \mathbf{R} between the angular velocity vector of the end-effector and the components of $\dot{\mathbf{i}}$ relative to the end-effector orientation; since the expression of \mathbf{R} depends on the adopted minimal representation of the orientation, a configuration at which \mathbf{T} is singular is then referred to as a *representation singularity*. Remarkably, any minimal description of the end-effector orientation experiences the occurrence of representation singularities. Furthermore, a given configuration may or may not yield a representation singularity depending on which description of orientation is used.

A representation singularity is not directly related to the true motion capabilities of the manipulator structure, which can instead be inferred by the analysis of the geometric Jacobian matrix. Rank deficiencies of \mathbf{J} are in fact related to loss of mobility of the manipulator end-effector; indeed, end-effector velocities exist in this case which are unfeasible for any velocity commanded at the joints. A configuration at which \mathbf{J} is singular is referred to as a *kinematic singularity*.

Since this chapter focuses on the inversion of the task differential kinematics (11.2) and (11.8), in the following the task Jacobian matrix and its singularities

(which include representation and kinematic singularities) are studied in detail. The case $N \geq M$ is considered, which encompasses conventional as well as redundant manipulators.

Singular Value Decomposition of the Jacobian

To analyze the linear mapping (11.2), the *singular value decomposition* (SVD) of the Jacobian matrix is adopted; remarkably, this powerful numerical tool is the sole reliable method to compute the rank of a matrix and to study near-singular linear mappings. The classic *Golub–Reinsch* algorithm [11.2], which is the most efficient and numerically stable algorithm to compute the SVD of an arbitrary matrix, may however be computationally demanding in view of real-time applications. A faster algorithm that takes advantage of the nature of robotic matrix calculations has been proposed [11.3]; this makes it possible to improve real-time kinematic control techniques.

The SVD of the task Jacobian matrix can be written in the form

$$\mathbf{J}_t = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \sum_{i=1}^M \sigma_i \mathbf{u}_i \mathbf{v}_i^T, \quad (11.9)$$

where \mathbf{U} is the $M \times M$ orthonormal matrix of the output singular vectors \mathbf{u}_i , \mathbf{V} is the $N \times N$ orthonormal matrix of the input singular vectors \mathbf{v}_i , and $\mathbf{\Sigma} = (\mathbf{S} \mathbf{0})$ is the $M \times N$ matrix whose $M \times M$ diagonal submatrix \mathbf{S} contains the singular values σ_i of the matrix \mathbf{J}_t . It is worth noticing that the SVD is a continuous and well-behaved function of its matrix argument; therefore, the input and output singular vectors as well as the singular values do not change much in the neighborhood of the current configuration. Letting $\text{rank}(\mathbf{J}_t) = R$, the following hold:

- $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_R > \sigma_{R+1} = \dots = 0$;
- $\mathcal{R}(\mathbf{J}_t) = \text{span}\{\mathbf{u}_1, \dots, \mathbf{u}_R\}$;
- $\mathcal{N}(\mathbf{J}_t) = \text{span}\{\mathbf{v}_{R+1}, \dots, \mathbf{v}_N\}$.

If the task Jacobian is full-rank ($R=M$), all the singular values are nonzero, the range space of \mathbf{J}_t is the entire \mathbb{R}^M , and the null space of \mathbf{J}_t has dimension $N-M$. In a singular configuration, instead, it is $R < M$; thus, the last $M-R$ singular values are zero, the range space of \mathbf{J}_t is an R -dimensional subspace of \mathbb{R}^M , and the dimension of the null space of \mathbf{J}_t increases to $N-R$. An interpretation of this from a kinematic viewpoint is presented in the following.

Feasible Velocities. At each configuration, the range space of \mathbf{J}_t is the set of task-space velocities that can

be obtained as a result of all possible joint-space velocities $\dot{\mathbf{q}}$; therefore, it constitutes the so-called subspace of *feasible velocities* for the end-effector task. A base of $\mathcal{R}(\mathbf{J}_t)$ is given by the first R output singular vectors, which represent independent linear combinations of the single components of the task velocities. Accordingly, the effect of a singularity is to decrease the dimension of the range space of \mathbf{J}_t by eliminating a linear combination of task velocity components from the space of feasible velocities.

Null-Space Velocities. At each configuration, the null space of \mathbf{J}_t is the set of joint-space velocities which yield zero task velocity; these are thus shortly called *null-space velocities*. A base of $\mathcal{N}(\mathbf{J}_t)$ is given by the $N-R$ last input singular vectors, which represent independent linear combinations of the velocities at each joint. From this viewpoint, the effect of a singularity is to increase the dimension of the null space of \mathbf{J}_t by introducing a further independent linear combination of joint velocities which produces a zero task velocity.

According to (11.2) and (11.9), a joint velocity along the i -th input singular vector results in a task velocity which lies along the i -th output singular vector:

$$\forall \rho \in \mathbb{R} \quad \dot{\mathbf{q}} = \rho \mathbf{v}_i \quad \Rightarrow \quad \mathbf{t} = \sigma_i \rho \mathbf{u}_i.$$

Thus, the i -th singular value of \mathbf{J}_t can be viewed as a gain factor relating motion along the \mathbf{v}_i direction of the joint velocity space to the resulting motion along the \mathbf{u}_i direction of the task velocity space. When a singularity is approached, the R -th singular value σ_R tends to zero and the task velocity produced by a fixed joint velocity along \mathbf{v}_R is decreased proportionally. At a singular configuration, the joint-space velocity along \mathbf{v}_R belongs to the null-space velocities and task velocities along \mathbf{u}_R become unfeasible for the manipulator.

In the generic case, the joint-space velocity $\dot{\mathbf{q}}$ is an arbitrary linear combination of individual joint velocities with nonzero components along all the \mathbf{v}_i . Its effect can be analyzed by combining the single effects of the above components. Remarkably, the components of $\dot{\mathbf{q}}$ in the null space of \mathbf{J}_t produce a change in the configuration of the manipulator without affecting its task velocity. This can be exploited to achieve additional goals – such as obstacle or singularity avoidance – in addition to the realization of a desired task motion, and constitutes the core of redundancy resolution approaches.

Distance from Singularities

Clearly, the effect of a singularity is experienced not only at the singular configuration itself but also in its

neighborhood (see also Sect. 11.3.2). For this reason, it is important to be able to characterize the distance from singularities through suitable measures; these can then be exploited to counteract undesirable effects.

Since each singularity is associated to a rank loss of \mathbf{J}_t , one conceptually simple possibility in the case of a square Jacobian matrix ($M=N$) is to compute its determinant. A generalization of this idea that works also for nonsquare Jacobians is the *manipulability measure* [11.4], defined as

$$\mu = \sqrt{|\mathbf{J}_t \mathbf{J}_t^\top|}.$$

It can be recognized that the manipulability measure is equal to the product of the singular values of \mathbf{J}_t , i. e.,

$$\mu = \prod_{i=1}^M \sigma_i,$$

and thus its zeros coincide with the singularities.

Another possible measure of distance from a singular configuration is the *condition number* of the Jacobian matrix [11.5], defined as

$$\kappa = \frac{\sigma_1}{\sigma_M}.$$

The condition number has values ranging from 1, at configurations in which all the singular values are equal,

to ∞ , at singular configurations. Note that when $\kappa=1$ all the singular values are equal and thus the end-effector has the same motion capability in all task space directions – i. e., the arm is at an *isotropic configuration* – whereas at a singularity it loses mobility in some task-space direction.

An even more direct measure of the distance from singular configurations is the *smallest singular value* of the Jacobian matrix [11.5], i. e.,

$$\sigma_{\min} = \sigma_M.$$

A computationally light estimate of the smallest singular value can be obtained either via numerical methods [11.3, 6, 7] or based on a kinematic analysis of the robot structure [11.8].

It must be noted that the manipulability measure may remain constant even in the presence of significant variations of either the condition number or the smallest singular value of \mathbf{J}_t . On the other hand, since the smallest singular value changes more radically near singularities than the other singular values, it dominates there the behavior of the determinant and the condition number of the Jacobian matrix; therefore, the most effective measure of distance from singular configurations is the smallest singular value of \mathbf{J}_t [11.5].

11.3 Inverse Differential Kinematics

In order to accomplish a task, a proper joint motion must be commanded to the manipulator; therefore, it is necessary to derive mathematical relations which allow to compute joint-space variables corresponding to the assigned task-space variables. This is the objective of the inverse kinematics problem.

The inverse kinematics problem can be solved by inverting either the direct kinematics equation (11.1), the first-order differential kinematics (11.2) or the second-order differential kinematics (11.8). If the task is time-varying (i. e., if a desired trajectory $\mathbf{t}(t)$ is assigned), it is convenient to solve the differential kinematic relationships because these represent linear equations with the task Jacobian as the coefficient matrix [11.9].

11.3.1 The General Solution

Under the assumption that the manipulator is kinematically redundant (i. e., $M < N$), the general solution of (11.2) or (11.8) can be expressed by resorting to the pseudoinverse \mathbf{J}_t^\dagger of the task Jacobian matrix [11.1, 10];

this is the unique matrix satisfying the Moore–Penrose conditions [11.11–13]

$$\begin{aligned} \mathbf{J}_t \mathbf{J}_t^\dagger \mathbf{J}_t &= \mathbf{J}_t \\ \mathbf{J}_t^\dagger \mathbf{J}_t \mathbf{J}_t^\dagger &= \mathbf{J}_t^\dagger \\ (\mathbf{J}_t \mathbf{J}_t^\dagger)^\top &= \mathbf{J}_t \mathbf{J}_t^\dagger \\ (\mathbf{J}_t^\dagger \mathbf{J}_t)^\top &= \mathbf{J}_t^\dagger \mathbf{J}_t. \end{aligned} \quad (11.10)$$

If \mathbf{J}_t is low-rectangular and full-rank, its pseudoinverse can be computed as

$$\mathbf{J}_t^\dagger = \mathbf{J}_t^\top (\mathbf{J}_t \mathbf{J}_t^\top)^{-1}. \quad (11.11)$$

If \mathbf{J}_t is square, expression (11.11) reduces to the standard inverse matrix.

The general solution of (11.2) can be written as

$$\dot{\mathbf{q}} = \mathbf{J}_t^\dagger \dot{\mathbf{t}} + (\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t) \dot{\mathbf{q}}_0. \quad (11.12)$$

Here, $\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t$ represents the orthogonal projection matrix in the null space of \mathbf{J}_t , and $\dot{\mathbf{q}}_0$ is an arbitrary

joint-space velocity; the second part of the solution is therefore a null-space velocity. Equation (11.12) provides all least-squares solution to the end-effector task constraint (11.2), i.e., it minimizes $\|\dot{\mathbf{i}} - \mathbf{J}\dot{\mathbf{q}}\|$. In particular, if \mathbf{J}_t is low-rectangular and full-rank, all joint velocities in the form (11.12) exactly realize the assigned task velocity. By acting on $\dot{\mathbf{q}}_0$, one can still obtain different joint velocities that give the same end-effector task velocity; therefore, as will be discussed later in detail, the solution in the form (11.12) is typically used in the context of redundancy resolution.

The particular solution obtained by setting $\dot{\mathbf{q}}_0 = 0$ in (11.12)

$$\dot{\mathbf{q}} = \mathbf{J}_t^\dagger \dot{\mathbf{i}} \quad (11.13)$$

provides the least-squares solution of (11.2) with minimum norm, and is known as the *pseudoinverse solution*. In terms of the inverse differential kinematics problem, the least-squares property quantifies the accuracy of the end-effector task realization, while the minimum norm property may be relevant for the feasibility of the joint-space velocities.

As for the second-order kinematics (11.8), its least-squares solutions can be expressed in the general form

$$\ddot{\mathbf{q}} = \mathbf{J}_t^\dagger (\ddot{\mathbf{i}} - \dot{\mathbf{J}}_t \dot{\mathbf{q}}) + (\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t) \ddot{\mathbf{q}}_0, \quad (11.14)$$

where $\ddot{\mathbf{q}}_0$ is an arbitrary joint-space acceleration. As above, choosing $\ddot{\mathbf{q}}_0 = 0$ in (11.14) gives the minimum-norm acceleration solution:

$$\ddot{\mathbf{q}} = \mathbf{J}_t^\dagger (\ddot{\mathbf{i}} - \dot{\mathbf{J}}_t \dot{\mathbf{q}}). \quad (11.15)$$

11.3.2 Singularity Robustness

We now investigate the kinematics aspects involved by the first-order inverse mappings (11.12) and (11.13) with respect to the handling of singularities. With reference to the SVD of \mathbf{J}_t in (11.9), let us consider the following decomposition of the matrix \mathbf{J}_t^\dagger

$$\mathbf{J}_t^\dagger = \mathbf{V} \boldsymbol{\Sigma}^\dagger \mathbf{U}^\top = \sum_{i=1}^R \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{u}_i^\top, \quad (11.16)$$

where, as above, R denotes the rank of the task Jacobian matrix. Analogously to (11.9), the following hold:

- $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_R > \sigma_{R+1} = \dots = 0$;
- $\mathcal{R}(\mathbf{J}_t^\dagger) = \mathcal{N}^\perp(\mathbf{J}_t) = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_R\}$;
- $\mathcal{N}(\mathbf{J}_t^\dagger) = \mathcal{R}^\perp(\mathbf{J}_t) = \text{span}\{\mathbf{u}_{R+1}, \dots, \mathbf{u}_M\}$.

Notice that, if the Jacobian matrix is full-rank, the range space of \mathbf{J}_t^\dagger is an M -dimensional subspace of \mathbb{R}^N and the null space of \mathbf{J}_t^\dagger is trivial. In a singular configuration ($R < M$), instead, the range space of \mathbf{J}_t^\dagger is an R -dimensional subspace of \mathbb{R}^N , and an $M-R$ -dimensional null space of \mathbf{J}_t^\dagger exists.

The range space of \mathbf{J}_t^\dagger is the set of joint-space velocities $\dot{\mathbf{q}}$ that can be obtained via the inverse kinematic mapping (11.13) as a result of all possible task velocities $\dot{\mathbf{i}}$. Since these $\dot{\mathbf{q}}$ belong to the orthogonal complement of the null space of \mathbf{J}_t , the pseudoinverse solution (11.13) satisfies the least-squares condition, as expected.

The null space of \mathbf{J}_t^\dagger is the set of the task velocities $\dot{\mathbf{i}}$ which yield null joint-space velocity in the current configuration; these $\dot{\mathbf{i}}$, on the other hand, belong to the orthogonal complement of the space of feasible task velocities. Therefore, one effect of the pseudoinverse solution (11.13) is to filter out the unfeasible components of the commanded task velocity while allowing exact tracking of the feasible components; this is related to the minimum-norm property.

If the assigned task velocity is aligned with \mathbf{u}_i , the corresponding joint-space velocity – computed via (11.13) – is obtained along \mathbf{v}_i modulated by the factor $1/\sigma_i$. When a singularity is approached, the R -th singular value tends to zero and a fixed task velocity command along \mathbf{u}_R requires joint-space velocities along \mathbf{v}_R that grow unboundedly in proportion to the factor $1/\sigma_R$. At the singular configuration, the \mathbf{u}_R direction becomes unfeasible for the task variables and \mathbf{v}_R adds to the null-space velocities of the arm.

According to the above, two main problems are inherently related to the basic inverse differential kinematics solution (11.13), namely:

- at near-singular configurations, excessive joint-space velocities may result, due to the component of $\dot{\mathbf{i}}$ along the direction which becomes unfeasible at the singularity;
- at the singular configuration, discontinuity of the joint-space solution is experienced if $\dot{\mathbf{i}}$ has a non-null unfeasible component.

The same is obviously true for the complete inverse solution (11.12).

Both the above problems are of major concern for kinematic control of manipulators, where the computed joint-space velocities must actually be executed by the robot arm. This has motivated the development of modified inverse differential mappings, so as to en-

sure proper behavior of the manipulator throughout its workspace. A reasonable approach is to preserve the mapping (11.13) far from singularities and to set up local modifications of it only inside a region enclosing the singular configuration; the definition of the region depends on a suitable measure of distance from the singularity, while the modified mapping must ensure feasible and continuous joint velocities.

Modification of the Planned Trajectory

One way to tackle the problem of singularities is to act at the planning level by either designing trajectories that avoid unavoidable singularities or assigning task-space motion commands that are feasible to the robot arm. However, these solutions rely on perfect task planning and cannot be used in real-time sensory control applications, in which motion commands are generated online.

Avoidance of singular configurations at the motion planning level is relatively simple when dealing with those naturally characterized in the task space such as the shoulder singularity of an anthropomorphic arm. However, this approach may result difficult for those singularities, like the wrist one, that may occur everywhere in the workspace.

Another possibility is to perform a joint-space interpolation when the planned trajectory is close to a singularity [11.14]; nevertheless, this may cause large errors in tracking the originally assigned task-space motion.

Acting in the task space, a method based on time-scale transformation is presented in [11.15], which slows down the manipulator's motion when a singularity is approached; this technique, however, fails at the singularity.

Since a task-space robot control system must be able to guide the motion of the manipulator safely through singularities, considerable research effort has been instead devoted to the derivation of well-defined and continuous inverse kinematic mappings.

Removal of Dependent Rows or Columns of the Jacobian Matrix

The first requirement of solution (11.13) is the availability of a general algorithm to compute the pseudoinverse of \mathbf{J}_t also when the Jacobian is singular. Several techniques presented in the literature can be arranged in this framework; they consist either in removing the unfeasible end-effector reference components [11.16] or in using nonsingular blocks of the Jacobian matrix [11.10]. The main problem with this type of solution is the spec-

ification of the directions of unfeasible velocities in a systematic way and the need to have smooth transitions between the usual inverse kinematic algorithm and the algorithms used close to the singularities.

A systematic procedure to compute the pseudoinverse of the Jacobian matrix can be devised by taking advantage of kinematic analysis of the manipulator structure since for typical manipulators it is possible to identify and describe classes of singular configurations in suitably defined link-fixed frames. The approach has been demonstrated for a six-degree-of-freedom elbow geometry in [11.17, 18].

As for the continuity of the solution across a singularity, it must be observed that, while the singular vectors change very little in the neighborhood of the singularity, the term

$$\frac{1}{\sigma_M} \mathbf{v}_M \mathbf{u}_M^\top \dot{\mathbf{t}}$$

suddenly disappears from (11.16) when R becomes less than M at the singular configuration.

One possibility to avoid this problem is to make

$$\mathbf{u}_M^\top \dot{\mathbf{t}} \approx 0$$

in the neighborhood of the singularity, which means avoiding commanding task velocities along the direction that becomes unfeasible at the singular configuration. Nevertheless, as anticipated Sect. 11.3.2, it is reasonable to apply this approach only for preplanned trajectories in relation to unavoidable singularities that are naturally characterized in the task space.

Independently from the assigned $\dot{\mathbf{t}}$, continuity of the pseudoinverse solution can be ensured by treating the manipulator as singular in a suitably defined region around each singularity through a modified Jacobian $\bar{\mathbf{J}}_t$, so that $M-R$ extra degrees of freedom are available [11.17–19]; these can be used without significantly affecting the end-effector velocity, because the modified Jacobian $\bar{\mathbf{J}}_t$ approximates \mathbf{J}_t inside the region. The described approach is difficult to generalize for multiple singularities. For the typical anthropomorphic geometry of industrial manipulators, however, only the wrist singularity is of primary concern since it may occur everywhere in the workspace, while the elbow and shoulder singularities are naturally characterized in the task space and thus can be avoided during planning.

Regularization/Damped Least-Squares Technique

The use of the damped least-squares technique in the inverse differential kinematics problem has been

independently proposed in [11.9, 20]. The method corresponds to solving the equation

$$\mathbf{J}_t^\top \dot{\mathbf{i}} = (\mathbf{J}_t^\top \mathbf{J}_t + \lambda^2 \mathbf{I}) \dot{\mathbf{q}} \quad (11.17)$$

in place of (11.2). Here, $\lambda \in \mathbb{R}$ is the damping factor. It can be recognized that, when λ is zero, (11.17) and (11.2) become identical.

The solution to (11.17) can be written in two equivalent forms:

$$\dot{\mathbf{q}} = \mathbf{J}_t^\top (\mathbf{J}_t \mathbf{J}_t^\top + \lambda^2 \mathbf{I})^{-1} \dot{\mathbf{i}}, \quad (11.18)$$

$$\dot{\mathbf{q}} = (\mathbf{J}_t^\top \mathbf{J}_t + \lambda^2 \mathbf{I})^{-1} \mathbf{J}_t^\top \dot{\mathbf{i}}. \quad (11.19)$$

The computational load of (11.18) is lower than that of (11.19), since usually $N \geq M$. In the remainder, we will refer to the damped least-squares solution as

$$\dot{\mathbf{q}} = \mathbf{J}_t^*(\mathbf{q}) \dot{\mathbf{i}}, \quad (11.20)$$

whenever explicit specification of the computation used is not essential.

Solution (11.20) satisfies the condition

$$\min_{\dot{\mathbf{q}}} \left(\|\dot{\mathbf{i}} - \mathbf{J}_t \dot{\mathbf{q}}\|^2 + \lambda^2 \|\dot{\mathbf{q}}\|^2 \right), \quad (11.21)$$

which realizes a trade-off between the least-squares and the minimum-norm properties. Condition (11.21) implies consideration of accuracy and feasibility at the same time as choosing the joint-space velocity required to match the given $\dot{\mathbf{i}}$. In this regard it is essential to suitably select the value to be assigned to the damping factor: small values of λ give accurate solution but low robustness to the occurrence of singular and near-singular configurations; high values of λ result in low tracking accuracy even when a feasible and accurate solution would be possible.

In the framework of singular value decomposition, the solution (11.20) can be written as

$$\dot{\mathbf{q}} = \sum_{i=1}^R \frac{\sigma_i}{\sigma_i^2 + \lambda^2} \mathbf{v}_i \mathbf{u}_i^\top \dot{\mathbf{i}}. \quad (11.22)$$

Remarkably, we have

- $\mathcal{R}(\mathbf{J}_t^*) = \mathcal{R}(\mathbf{J}_t^\dagger) = \mathcal{N}^\perp(\mathbf{J}_t) = \text{span}\{\mathbf{v}_1, \dots, \mathbf{v}_R\}$,
- $\mathcal{N}(\mathbf{J}_t^*) = \mathcal{N}(\mathbf{J}_t^\dagger) = \mathcal{R}^\perp(\mathbf{J}_t) = \text{span}\{\mathbf{u}_{R+1}, \dots, \mathbf{u}_M\}$.

Analogously to \mathbf{J}_t^\dagger , if the Jacobian matrix is full-rank, the range space of \mathbf{J}_t^* is an M -dimensional subspace of \mathbb{R}^N and the null space of \mathbf{J}_t^* is trivial; in a singular configuration, instead, the range space of \mathbf{J}_t^* is an R -dimensional subspace of \mathbb{R}^N , and an $M-R$ -dimensional null space of \mathbf{J}_t^* exists.

It is clear that, with respect to the pure least-squares solution (11.13), in (11.22) the components for which $\sigma_i \gg \lambda$ are little influenced by the damping factor, since in this case

$$\frac{\sigma_i}{\sigma_i^2 + \lambda^2} \approx \frac{1}{\sigma_i}.$$

On the other hand, when a singularity is approached, the smallest singular value tends to zero while the associated component of the solution is driven to zero by the factor σ_i/λ^2 ; this progressively reduces the joint velocity required to achieve near-degenerate components of the commanded $\dot{\mathbf{i}}$. At the singularity, the solutions (11.20) and (11.13) behave identically as long as the remaining singular values are significantly larger than the damping factor. Remarkably, due to the damping factor, an upper bound of $1/(2\lambda)$ exists on the gain factor relating for each i the task velocity component along \mathbf{u}_i to the resulting joint velocity along \mathbf{v}_i ; this bound is reached when $\sigma_i = \lambda$.

Selection of the Damping Factor. According to above, the value selected for the damping factor establishes the closeness of the current configuration to a singularity; moreover, λ determines the degree of approximation introduced with respect to the pure least-squares solution given by the pseudoinverse. An optimal choice for λ requires consideration of the smallest non-null singular value experienced along the given trajectory and of the minimum damping needed to ensure feasible joint velocities.

To achieve good performance in the entire manipulator's workspace the use of a configuration-varying damping factor has been proposed [11.9]. The natural choice is to adjust λ as a function of some measure of distance from the singularity at the current configuration of the robot arm. Far from singular configurations feasible joint velocities are obtained; thus the accuracy requirement prevails and low damping must be used. Close to a singularity, task velocity commands along the unfeasible directions give large joint velocities and the accuracy requirement must be relaxed; in these cases, high damping is needed.

A first proposal was to adjust the damping factor as a function of the manipulability measure [11.9]. Since a more effective measure of distance from singular configurations is the smallest singular value of the Jacobian matrix [11.5], its use has been considered later for building a variable damping factor.

If an estimate $\hat{\sigma}_M$ of the smallest singular value is available, the following choice for the damping factor

can be adopted [11.21]

$$\lambda^2 = \begin{cases} 0 & \text{when } \hat{\sigma}_M \geq \varepsilon \\ \left[1 - (\hat{\sigma}_M/\varepsilon)^2\right] \lambda_{\max}^2 & \text{otherwise,} \end{cases} \quad (11.23)$$

which ensures continuity and good shaping of the solution. In (11.23), ε defines the size of the singular region, in which damping is used, and λ_{\max} sets the maximum value of the damping factor, which occurs just at the singularity.

Numerical Filtering. As can be recognized from (11.22), the damping factor affects the accuracy of the solution along each end-effector velocity component; however, the sole unfeasible direction is responsible for the loss of tracking ability. To overcome this problem, selective filtering of the end-effector velocity components was proposed in [11.6]. The method can be generalized as follows; if an estimate $\hat{\mathbf{u}}_i$ of the output singular vector associated to the $M-K$ smallest singular values – which give the components to become unfeasible – is available, the solution can be written in the form

$$\dot{\mathbf{q}} = \mathbf{J}_t^\top \left(\mathbf{J}_t \mathbf{J}_t^\top + \lambda^2 \mathbf{I} + \beta^2 \sum_{i=K+1}^M \hat{\mathbf{u}}_i \hat{\mathbf{u}}_i^\top \right)^{-1} \mathbf{i}, \quad (11.24)$$

where β provides the largest contribution to damping only along the unfeasible components. This can also be recognized from the expression

$$\dot{\mathbf{q}} \approx \sum_{i=1}^K \frac{\sigma_i}{\sigma_i^2 + \lambda^2} \mathbf{v}_i \mathbf{u}_i^\top \mathbf{i} + \sum_{i=K+1}^R \frac{\sigma_i}{\sigma_i^2 + \lambda^2 + \beta^2} \mathbf{v}_i \mathbf{u}_i^\top \mathbf{i}, \quad (11.25)$$

in which the approximation is due to the estimates $\hat{\mathbf{u}}_i$ in (11.24). Notice that $K \leq R$; nevertheless, a nonzero value of λ is kept to guarantee satisfactory conditioning of the mapping (11.24) even for incorrect estimates of the output singular vectors.

Similarly, additional damping can be provided to the joint-space velocity components given by the input singular vectors associated to the smallest singular values, since they are close to become the null-space velocities [11.22]. The solution can be generalized in the form

$$\dot{\mathbf{q}} = \left(\mathbf{J}_t^\top \mathbf{J}_t + \lambda^2 \mathbf{I} + \beta^2 \sum_{i=K+1}^N \hat{\mathbf{v}}_i \hat{\mathbf{v}}_i^\top \right)^{-1} \mathbf{J}_t^\top \mathbf{i}, \quad (11.26)$$

where β provides the largest contribution to damping only along the $N-K$ estimated null-space velocity components $\hat{\mathbf{v}}_i$. This can be also recognized from the expression

$$\dot{\mathbf{q}} \approx \sum_{i=1}^K \frac{\sigma_i}{\sigma_i^2 + \lambda^2} \mathbf{v}_i \mathbf{u}_i^\top \mathbf{i} + \sum_{i=K+1}^R \frac{\sigma_i}{\sigma_i^2 + \lambda^2 + \beta^2} \mathbf{v}_i \mathbf{u}_i^\top \mathbf{i}, \quad (11.27)$$

in which again the approximation is due to the estimates $\hat{\mathbf{v}}_i$ in (11.26). Also in this case a nonzero value of λ is kept to guarantee satisfactory conditioning of the mapping (11.26) even for incorrect estimates of the input singular vectors.

Comparison of (11.27) with (11.25) shows that, in the case of accurate estimate of the singular vectors, the solutions (11.24) and (11.26) behave identically.

11.3.3 Joint Trajectory Reconstruction

When solving the inverse kinematics problem at the first-order differential level, one obtains the joint velocity profile $\dot{\mathbf{q}}(t)$ that corresponds to an assigned end-effector task velocity profile $\mathbf{i}(t)$; however, the robot motion controller needs reference position trajectories in addition to the velocity references for the joints. The reconstruction of the joint-position profile from the joint-velocity profile obtained from an end-effector task velocity profile $\mathbf{i}(t)$ realizes a kinematic inversion that can be seen as an *inverse kinematics algorithm*.

If the joint-velocity profile were completely specified (e.g., by its analytical expression) the corresponding joint-position profile could be obtained from time integration

$$\mathbf{q}(t) = \mathbf{q}(t_0) + \int_{t_0}^t \dot{\mathbf{q}}(\tau) d\tau. \quad (11.28)$$

Nevertheless, digital implementation of the robot control system makes it more likely that a discrete-time sequence of samples $\dot{\mathbf{q}}_k$ of the computed joint velocities at the time instants t_k will be available, i.e.,

$$\dot{\mathbf{q}}_k = \dot{\mathbf{q}}(t_k).$$

For this reason, a discrete-time numerical approximation of the continuous-time integral (11.28) must be suitably devised.

Accurate discrete-time approximations of a continuous-time integral usually require a trade-off between the complexity of the interpolating algorithm and the

length of the time step. In real-time applications, such as robot motion control, high-order interpolation results in large finite-time delay that degrade the dynamic performance of the control loop. This time delay can be reduced by suitably shortening the time step; however, if the time step is sufficiently short even a low-order interpolation may give acceptable accuracy of the numerical integration. The typical case is that of a first-order interpolation, e.g., an Euler forward rectangular rule that transforms the integral (11.28) into

$$\mathbf{q}_k = \mathbf{q}_0 + \sum_{h=0}^{k-1} \dot{\mathbf{q}}_h \Delta t, \quad (11.29)$$

where Δt is the time step. Equation (11.29) is usually written in the more effective recursive form

$$\mathbf{q}_k = \mathbf{q}_{k-1} + \dot{\mathbf{q}}_{k-1} \Delta t.$$

Closed-Loop Inverse Kinematics (CLIK)

No matter what kind of interpolation is used, the small though unavoidable error suffered at each numerical integration step accumulates, leading to long-term drifting

of the reconstructed profile from the exact joint-position profile. Another source of error affecting any integral reconstruction method is the possible uncertainty in the initial value of the joint position.

Algorithmic solutions that overcome these problems are based on the use of a feedback correction term; these are termed *closed-loop inverse kinematics* (CLIK) algorithms [11.23]. Considering, e.g., the case of first-order kinematics, the joint velocity at the k -th time instant is computed as (compare with (11.12))

$$\begin{aligned} \dot{\mathbf{q}}_k = & \mathbf{J}_t^\dagger(\mathbf{q}_k) \{ \dot{\mathbf{t}}_k + \mathbf{K}[\mathbf{t}_k - \mathbf{k}_t(\mathbf{q}_k)] \} \\ & + [\mathbf{I} - \mathbf{J}_t^\dagger(\mathbf{q}_k) \mathbf{J}_t(\mathbf{q}_k)] \dot{\mathbf{q}}_{0k}, \end{aligned} \quad (11.30)$$

where \mathbf{K} is a constant positive-definite gain matrix.

Second-order CLIK algorithms are also available that solve for joint positions, velocities, and accelerations [11.24, 25]. The CLIK approach was originally proposed in [11.26] and [11.27] based on the Jacobian transpose in lieu of the pseudoinverse, which provides remarkable computational savings and provides inherent singularity handling capabilities [11.28].

11.4 Redundancy Resolution via Optimization

For a kinematically redundant manipulator, the inverse kinematics problem admits an infinite number of solutions, so that a criterion to select one of them is needed. In this section we will consider the problem of redundancy resolution via optimization at the first-order differential kinematics level. Before discussing algorithmic schemes for computing joint velocities, we provide a short review of possible performance criteria.

11.4.1 Performance Criteria

The availability of degrees of freedom in excess with respect to those strictly needed to execute a given task can be used to improve the value of performance criteria during the motion. These criteria may depend on the robot joint configuration only, or also involve velocities and/or accelerations.

Among the additional objectives that can be pursued by defining a suitable criterion, the most important is probably singularity avoidance. In fact, one of the main reasons for introducing kinematic redundancy is to reduce the extension of the workspace region where the manipulator is necessarily a singular configuration (*unavoidable* singularities). A discussion of avoidable and unavoidable singularities in redundant manipula-

tors can be found in [11.29]. If the assigned end-effector task does not pass through unavoidable singularities, it is in principle always possible to compute a joint trajectory along which the task Jacobian \mathbf{J}_t is continuously full-rank. To this end, possible performance criteria are the configuration-dependent functions introduced in Sect. 11.2.2 that characterize the distance from singularities, i.e., the manipulability measure, the condition number, and the smallest singular value of \mathbf{J}_t . Maximizing (or keeping as large as possible) these functions during the motion is a reasonable plan in order to avoid singular configurations during the motion.

Since kinematic inversion produces diverging joint velocities in the vicinity of singular configurations, a conceptually different possibility is to minimize the norm of the joint velocity generated by the redundancy resolution scheme. However, this approach guarantees singularity avoidance only if such a norm is minimized in an integral sense along the motion of the manipulator. Local minimization of the norm [11.10] does not lead to singularity avoidance in any practical sense [11.29].

Redundancy can be also used to keep the manipulator linkage away from undesired regions of the joint space, for example, mechanical joint limits that are typically present in robot manipulators may be avoided by

minimizing the cost function [11.30]

$$\mathbf{H}(\mathbf{q}) = \frac{1}{2} \sum_{i=1}^N \left(\frac{q_i - q_{i,\text{mid}}}{q_{i,\text{max}} - q_{i,\text{min}}} \right)^2$$

where $[q_{i,\text{min}}, q_{i,\text{max}}]$ is the available range for joint i and $q_{i,\text{mid}}$ is its midpoint. Another interesting application is obstacle avoidance, which can be enforced by minimizing suitable *artificial potential* functions defined on the basis of the image of the obstacle region in the configuration space [11.31, 32].

Many other performance criteria have been proposed in the literature; some of them are mentioned in Sects. 11.6 and 11.9.

11.4.2 Local Optimization

The simplest form of local optimization is represented by the pseudoinverse solution (11.13), which provides the joint velocity with the minimum norm among those which realize the task constraint. Clearly, the joint movement generated by this locally optimal solution does not provide global velocity minimization along the whole manipulator motion. This means that, despite the local minimization of joint velocities, singularity avoidance is not guaranteed [11.29].

Another possibility is to use the general solution (11.12), choosing the arbitrary joint velocity $\dot{\mathbf{q}}_0$ in the direction of the antigradient of a scalar configuration-dependent performance criteria $\mathbf{H}(\mathbf{q})$ which must be minimized:

$$\dot{\mathbf{q}}_0 = -k_{\mathbf{H}} \nabla \mathbf{H}(\mathbf{q}), \quad (11.31)$$

where $k_{\mathbf{H}}$ is a scalar step size and $\nabla \mathbf{H}(\mathbf{q})$ denotes the gradient of \mathbf{H} at the current joint configuration. This leads to the following redundancy resolution scheme [11.30]

$$\dot{\mathbf{q}} = \mathbf{J}_t^\dagger \dot{\mathbf{t}} - k_{\mathbf{H}} (\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t) \nabla \mathbf{H}(\mathbf{q}). \quad (11.32)$$

Since its second term is the projection of the antigradient of \mathbf{H} in the null space of the task Jacobian, the above expression is reminiscent of the projected gradient method for constrained minimization [11.33]. In particular, it can be shown [11.34] that the inverse kinematic solution (11.32) minimizes the complete quadratic function

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^\top \dot{\mathbf{q}} + k_{\mathbf{H}} \dot{\mathbf{q}}^\top \nabla \mathbf{H}(\mathbf{q})$$

at the current configuration \mathbf{q} . Thus, (11.32) represents a natural trade-off between the unconstrained local minimization of the performance criteria \mathbf{H} (which would lead to choose $\dot{\mathbf{q}} = -k_{\mathbf{H}} \nabla \mathbf{H}(\mathbf{q})$) and the satisfaction of constraint (11.2) by a minimum-norm joint velocity.

The choice of the step size $k_{\mathbf{H}}$ is critical for the performance of the redundancy resolution scheme (11.32). In particular, a small value of the step size may slow down the minimization of the performance criteria, but on the other hand a large value may even lead to an increase of \mathbf{H} (recall that the antigradient is the *local* direction of maximum decrease). In practice, to identify at each configuration an appropriate value of $k_{\mathbf{H}}$ in a reasonable time, one may use a simplified *line search* technique such as Armijo's rule [11.33].

11.4.3 Global Optimization

The main advantage of the redundancy resolution scheme (11.32) is its simplicity: if the computation of $\nabla \mathbf{H}(\mathbf{q})$ and $k_{\mathbf{H}}$ is efficient, such a scheme is a realistic option for real-time kinematic inversion. Its disadvantage is to be found in the local nature of the optimization process, which can lead to unsatisfactory performance over long tasks, for example, use of (11.32) with $\mathbf{H} = -\mu$ (the manipulability measure) will perform better than the simple pseudoinverse solution, but still cannot guarantee singularity avoidance.

It is therefore natural to consider the possibility of selecting $\dot{\mathbf{q}}_0$ in (11.12) so as to minimize integral criteria of the form

$$\int_{t_i}^{t_f} \mathbf{H}(\mathbf{q}) \, dt$$

defined over the duration $[t_i, t_f]$ of the whole task (e.g., the integral manipulability along the motion). Unfortunately, the solution of this problem (naturally formulated within the framework of calculus of variations) may not exist, and in any case it admits no closed form in general. Remarkably, one way to make the problem certainly solvable is to include under the integral a quadratic form in the joint velocities or accelerations. However, this is more easily done at the second-order kinematic level (see Sect. 11.6).

11.5 Redundancy Resolution via Task Augmentation

Another approach to redundancy resolution consists in augmenting the task vector so as to tackle ad-

ditional objectives expressed as constraints. In this section, we review the basic task augmentation tech-

niques for solving the first-order differential kinematics (11.2).

11.5.1 Extended Jacobian

The extended Jacobian technique, proposed by *Bailieul* [11.35] and later revisited by *Chang* [11.36], enforces an appropriate number of functional constraints to be fulfilled along with the original end-effector task so as to identify a single solution among the infinite compatible with the end-effector task.

Let us consider an objective function $g(\mathbf{q})$ to be optimized and let also $N_{J_t}(\mathbf{q})$ be a matrix spanning the null space of J_t at a nonsingular configuration \mathbf{q} , e.g.,

$$N_{J_t} = I - J_t^\dagger J_t.$$

It can be recognized that, for a given t_0 , if \mathbf{q}_0 is a configuration at which the function $g(\mathbf{q})$ is at an extreme under the constraint $t_0 = \mathbf{k}_t(\mathbf{q}_0)$, one has

$$\left. \frac{\partial g(\mathbf{q})}{\partial \mathbf{q}} \right|_{\mathbf{q}=\mathbf{q}_0} N_{J_t}(\mathbf{q}_0) = \mathbf{0}^\top. \quad (11.33)$$

If the Jacobian J_t has full rank M , then N_{J_t} has rank $N-M$; therefore, equation (11.33) yields a set of $N-M$ independent constraints that can be written in vector form as

$$\mathbf{h}(\mathbf{q}) = \mathbf{0};$$

for example, these can be obtained by taking the scalar product of the gradient $\partial g(\mathbf{q})/\partial \mathbf{q}$ by each of the $N-M$ vectors constituting a base of the null space of J_t , i. e.,

$$\mathbf{h}(\mathbf{q}) = \left(\frac{\partial g(\mathbf{q})}{\partial \mathbf{q}} (\mathbf{v}_{M+1}(\mathbf{q}) \dots \mathbf{v}_N(\mathbf{q})) \right)^\top.$$

At this point, the condition (11.33) implies that the equation

$$\begin{pmatrix} \mathbf{k}_t(\mathbf{q}_0) \\ \mathbf{h}(\mathbf{q}_0) \end{pmatrix} = \begin{pmatrix} t_0 \\ \mathbf{0} \end{pmatrix}$$

is satisfied.

For motion starting from t_0 with \mathbf{q}_0 that tracks a trajectory $\mathbf{t}(t)$ by keeping $g(\mathbf{q})$ extremized at each time one has

$$\begin{pmatrix} \mathbf{k}_t(\mathbf{q}(t)) \\ \mathbf{h}(\mathbf{q}(t)) \end{pmatrix} = \begin{pmatrix} \mathbf{t}(t) \\ \mathbf{0} \end{pmatrix}.$$

By differentiating both sides with respect to time one obtains

$$\begin{pmatrix} J_t(\mathbf{q}) \\ \frac{\partial \mathbf{h}(\mathbf{q})}{\partial \mathbf{q}} \end{pmatrix} \dot{\mathbf{q}} = \begin{pmatrix} \dot{\mathbf{t}} \\ \mathbf{0} \end{pmatrix}, \quad (11.34)$$

where the matrix premultiplying the vector $\dot{\mathbf{q}}$ is square and is called the *extended Jacobian* J_{ext} .

Therefore, if the initial configuration \mathbf{q}_0 extremizes $g(\mathbf{q})$ and provided that J_{ext} does not become singular, the time integral of the inverse mapping

$$\dot{\mathbf{q}} = J_{\text{ext}}^{-1}(\mathbf{q}) \begin{pmatrix} \dot{\mathbf{t}} \\ \mathbf{0} \end{pmatrix} \quad (11.35)$$

tracks the assigned end-effector trajectory $\mathbf{t}(t)$ propagating joint configurations that extremize $g(\mathbf{q})$.

The extended Jacobian method has a major advantage over pseudoinverse techniques of the form (11.13) in that it is *cyclic* (see Sect. 11.7). Moreover, solution (11.35) can be made equivalent to (11.12) via suitable choice of the vector $\dot{\mathbf{q}}_0$ [11.35, 37].

11.5.2 Augmented Jacobian

Another approach, the so-called task-space augmentation, introduces a constraint task to be fulfilled along with the end-effector task; then, an augmented Jacobian matrix is set-up whose inverse gives the sought joint velocity solution. The concept of task-space augmentation has been independently introduced by *Sciavicco* and *Siciliano* [11.28, 38, 39] and *Egeland* [11.40] and later revisited by *Seraji* in the framework of the configuration control method [11.41].

In detail, let us consider the vector $\mathbf{t}_c = (t_{c,1} \dots t_{c,p})^\top$ which describes the additional tasks to be fulfilled besides the M -dimensional end-effector task \mathbf{t} . In the general case $P \leq N-M$, although full redundancy exploitation suggests the consideration of exactly as many additional tasks as the number of redundant degrees of freedom, i. e., $P = N-M$.

The relation between the joint-space coordinate vector \mathbf{q} and the *constraint-task* vector \mathbf{t}_c can be considered as a direct kinematics equation

$$\mathbf{t}_c = \mathbf{k}_c(\mathbf{q}), \quad (11.36)$$

where \mathbf{k}_c is a continuous nonlinear vector function. Accordingly, it is useful to consider the mapping

$$\dot{\mathbf{t}}_c = J_c(\mathbf{q}) \dot{\mathbf{q}}, \quad (11.37)$$

that can be obtained by differentiating Eq. (11.36). In (11.37), $\dot{\mathbf{t}}_c$ is the constraint-task velocity vector, and $\mathbf{J}_c(\mathbf{q}) = \partial \mathbf{k}_c / \partial \mathbf{q}$ is the $P \times N$ constraint-task Jacobian matrix.

At this point, an *augmented-task* vector can be defined by stacking the end-effector task vector with the constraint-task vector as

$$\mathbf{t}_a = \begin{pmatrix} \mathbf{t} \\ \mathbf{t}_c \end{pmatrix} = \begin{pmatrix} \mathbf{k}_t(\mathbf{q}) \\ \mathbf{k}_c(\mathbf{q}) \end{pmatrix}.$$

According to this definition, finding a joint configuration \mathbf{q} that results in some desired value for \mathbf{t}_a means satisfying both the end-effector and the constraint task at the same time.

A solution to this problem can be found at the differential level by inverting the mapping

$$\dot{\mathbf{t}}_a = \mathbf{J}_a(\mathbf{q}) \dot{\mathbf{q}} \quad (11.38)$$

in which the matrix

$$\mathbf{J}_a = \begin{pmatrix} \mathbf{J}_t \\ \mathbf{J}_c \end{pmatrix}$$

is termed the *augmented Jacobian*.

A particular choice for the constraint-task vector is $\mathbf{t}_c = \mathbf{h}(\mathbf{q})$, with \mathbf{h} defined as explained in Sect. 11.5.1, which allows the augmented Jacobian method to embed the extended Jacobian one.

11.5.3 Algorithmic Singularities

The definition of additional goals besides tracking the end-effector velocity raises the possibility that configurations exist at which the augmented kinematics problem is singular while the sole end-effector task kinematics is not; these configurations are then termed *algorithmic singularities* [11.35]. With reference to the velocity mappings (11.34) and (11.38), an algorithmically singular configuration is one at which, respectively, the extended and the augmented Jacobians are singular while \mathbf{J}_t is full-rank.

Since the origin of task-augmentation redundancy resolution methods, *Baillieul* pointed out that algorithmic singularities are not a specific problem of the extended Jacobian technique but that they arise from the way in which the constraint task conflicts with the end-effector task [11.35, 37]. This is easily understandable in simple situations such as that of a trajectory tracking with obstacle avoidance problem: if the assigned trajectory passes through an obstacle either the trajectory is

tracked or the obstacle is avoided so that both tasks cannot be achieved. If the origin of the conflict between the two tasks has a clear meaning the algorithmic singularity may then be avoided by keenly specifying the constraint task case by case (e.g., [11.23]). In more general situations analytical tools may be useful in finding algorithmic singularities and guiding the choice of the constraint function [11.42] or in finding configurations that better harmonize the two tasks [11.43].

Looking at the tasks definitions in (11.2) and (11.37), by considering their inverse mappings it can be recognized that the two tasks are in conflict when

$$\mathcal{R}(\mathbf{J}_c^\top) \cap \mathcal{R}(\mathbf{J}_t^\top) \neq \{\mathbf{0}\} \quad (11.39)$$

and then this is the condition for the occurrence of an algorithmic singularity. On the other hand, when

$$\mathcal{R}(\mathbf{J}_c^\top) \cap \mathcal{R}(\mathbf{J}_t^\top) = \{\mathbf{0}\}$$

the two tasks are compatible since the two inverse mappings are linearly independent; a special case of task compatibility is when

$$\mathcal{R}(\mathbf{J}_c^\top) \equiv \mathcal{R}^\perp(\mathbf{J}_t^\top) \quad (11.40)$$

and the two mappings are orthogonal.

At the algorithmic singularities the augmented Jacobian cannot be inverted but singularity-robust techniques can be adopted. Since the exact solution does not exist there are reconstruction errors and these affect both the task vectors. To counteract this problem the use of weighted damped least-squares has been considered to invert the augmented Jacobian matrix [11.44, 45]. A different approach is the so-called task priority inverse kinematics.

11.5.4 Task Priority

Conflicts between the end-effector task and the constraint task are handled in the framework of the task-priority strategy by suitably assigning an order of priority to the given tasks and then satisfying the lower-priority task only in the null space of the higher-priority task [11.46, 47]. In the typical case, an end-effector task is considered as the primary task, although examples might be given in which it becomes the secondary task [11.23]. The idea is that, when an exact solution does not exist, the reconstruction error should only affect the lower-priority task.

With reference to solution (11.12), the task-priority method consists of computing $\dot{\mathbf{q}}_0$ so as to suitably achieve the P -dimensional constraint-task velocity $\dot{\mathbf{t}}_c$.

Remarkably, the projection of \dot{q}_0 onto the null space of J_t ensures lower priority of the constraint task with respect to the end-effector task since this results in a null-space velocity for it [11.48].

When the secondary task \dot{i}_c is orthogonal to the primary task \dot{i} (in the sense of Eq. (11.40)) the joint velocity

$$\dot{q}_0 = J_c^\dagger(q) \dot{i}_c \quad (11.41)$$

would easily solve the problem, being in addition already a null-space velocity for the primary-task velocity mapping (11.2) (i. e., projection through N_{J_t} would not be needed). However, in the general case the two tasks may be compatible but not orthogonal or may conflict and there not exist a joint velocity solution that ensures the achievement of both \dot{i} and \dot{i}_c . Coherently with the defined order of priority between the two tasks, a reasonable choice is then to guarantee exact tracking of the primary-task velocity while minimizing the constraint-task velocity reconstruction error $\dot{i}_c - J_c \dot{q}$; this gives [11.49]

$$\dot{q}_0 = [J_c(I - J_t^\dagger J_t)]^\dagger (\dot{i}_c - J_c J_t^\dagger \dot{i}). \quad (11.42)$$

Finally, by observing that the null-space projection operator is both hermitian and idempotent, the solution given by (11.12) and (11.42) can be simplified to [11.46]

$$\dot{q} = J_t^\dagger \dot{i} + [J_c(I - J_t^\dagger J_t)]^\dagger (\dot{i}_c - J_c J_t^\dagger \dot{i}). \quad (11.43)$$

It can be recognized that the problem of algorithmic singularities still remains; in fact, when condition (11.39) is satisfied the matrix $J_c(I - J_t^\dagger J_t)$ loses rank with full-rank J_t and J_c . However, differently from the task-space augmentation approach, correct primary-task solutions are expected as long as the sole primary-task Jacobian matrix is full-rank. On the other hand, out of the algorithmic singularities the task-priority strategy gives the same solution as the task-space augmentation approach; this implies that close

to an algorithmic singularity the solution becomes ill-conditioned and large joint velocities may result. This problem has partly been solved by looking at the efficient rank of the matrix involved and treating it as singular depending on a suitable threshold [11.46]; the obtained joint velocities are thus limited, but continuity of the null-space component must be ensured.

Another approach is to relax minimization of the secondary-task velocity reconstruction constraint and simply pursue tracking of the components of (11.41) that do not conflict with the primary task [11.50, 51], namely

$$\dot{q} = J_t^\dagger \dot{i} + (I - J_t^\dagger J_t) J_c^\dagger \dot{i}_c. \quad (11.44)$$

An intuitive justification of this solution can be given as follows: The pseudoinverses J_t^\dagger and J_c^\dagger are used to solve separately for the joint velocities associated with the respective task velocities; the joint velocity associated with the (secondary) constraint task is then projected onto the null space of J_t to remove the components that would interfere with the (primary) end-effector task and finally added to the joint velocity associated with the end-effector task. As a result, a nice property of solution (11.44) is that algorithmic singularities are decoupled from the singularities of J_c .

By construction, the solution (11.44) leads to larger constraint-task reconstruction errors than solution (11.43); this is the price paid to give smooth and feasible trajectories for the joint velocity in tracking conflicting tasks. Nevertheless, solution (11.44) is better used in a CLIK implementation which allows to recover the secondary-task tracking error out of the algorithmic singularities. In this case, it becomes

$$\dot{q} = J_t^\dagger w_t + (I - J_t^\dagger J_t) J_c^\dagger w_c$$

with

$$w_t = \dot{i} + K_t(t - k_t(q))$$

$$w_c = \dot{i}_c + K_c(t_c - k_c(q)).$$

11.6 Second-Order Redundancy Resolution

Redundancy resolution at the acceleration level allows the consideration of dynamic performance along the manipulator motion. Moreover, the obtained acceleration profiles can be directly used as reference signals (together with the corresponding positions and velocities) of a task-space dynamic controller. On the other hand, a second-order redundancy resolution scheme is

invariably more demanding in terms of computational load.

The simplest scheme operating at the acceleration level is represented by (11.15), i. e., the solution with the minimum norm among those which realize the task constraint (11.8). Similarly to the velocity-level pseudoinverse solution, the joint movement generated by

this locally optimal solution does not provide global acceleration minimization along the whole manipulator motion. Remarkably, however, the use of (11.15) leads to the minimization over $[t_i, t_f]$ of the integral index

$$\int_{t_i}^{t_f} \dot{\mathbf{q}}^\top \dot{\mathbf{q}} dt ,$$

provided that the appropriate boundary conditions are satisfied [11.52], for example, in the case of free endpoints (neither joint positions nor velocities specified at t_i and t_f) the boundary conditions to be satisfied are split and expressed as

$$\dot{\mathbf{q}}(\bar{t}) = \mathbf{J}_t^\dagger \dot{\mathbf{t}}(\bar{t}) \quad \bar{t} = t_i, t_f .$$

In spite of the apparent simplicity and elegance of the solution (11.15), therefore, actual minimization of the above integral cost requires the solution of a two-point boundary value problem (TPBVP), which is a computationally intensive numerical procedure impractical for real-time kinematic control. However, this may be perfectly acceptable for offline redundancy resolution in an industrial setting.

More flexibility in the choice of (both local and global) performance criteria is obviously obtained by considering the full second-order solution (11.14). Let the manipulator dynamic model be expressed as

$$\boldsymbol{\tau} = \mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_g(\mathbf{q}) , \quad (11.45)$$

where $\boldsymbol{\tau}$ is the vector of actuator torques, \mathbf{H} is the manipulator inertia matrix, \mathbf{c} is the vector of centrifugal/Coriolis terms, and $\boldsymbol{\tau}_g$ is the gravitational torque vector. Choosing the null-space acceleration in (11.14) as

$$\ddot{\mathbf{q}}_0 = -[\mathbf{H}(\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t)]^\dagger \ddot{\mathbf{t}} , \quad (11.46)$$

11.7 Cyclicity

A common drawback of redundancy resolution schemes based on differential kinematics is the lack of *cyclicity* (also called *repeatability*): in general, the joint-space trajectory corresponding to a cyclic task space trajectory is not cyclic itself (i. e., the final position of the joints does not coincide with the initial position). This phenomenon is clearly undesirable, because it basically means that the behavior of the manipulator along periodic tasks to be repeated over and over is unpredictable.

with

$$\ddot{\mathbf{t}} = \mathbf{H} \mathbf{J}_t^\dagger (\ddot{\mathbf{t}} - \dot{\mathbf{J}}_t \dot{\mathbf{q}}) + \mathbf{c} + \boldsymbol{\tau}_g .$$

leads to the local minimization of the actuator torque norm $\boldsymbol{\tau}^\top \boldsymbol{\tau}$ [11.53]. This particular redundancy resolution scheme performs reasonably over short tasks, but may lead to instability (more precisely, very high joint torques) in the long run, essentially due to the build-up of null-space joint velocities. Note also that the matrix product $\mathbf{H}(\mathbf{I} - \mathbf{J}_t^\dagger \mathbf{J}_t)$ in (11.46) is not full-rank, so that its pseudoinverse must be computed through a SVD procedure. Minimization of the integral joint torque is also possible [11.54]; this solution obviously avoids the instability problem, but once again the solution of a TPBVP is required.

Another interesting inverse solution is the following

$$\ddot{\mathbf{q}} = \mathbf{J}_{t,H}^\dagger (\ddot{\mathbf{t}} - \dot{\mathbf{J}}_t \dot{\mathbf{q}}) + (\mathbf{I} - \mathbf{J}_{t,H}^\dagger \mathbf{J}_t) \mathbf{H}^{-1} \mathbf{c} , \quad (11.47)$$

which is slightly modified with respect to the general formula (11.14) in view of the use of a weighted pseudoinverse. In particular, $\mathbf{J}_{t,H}^\dagger$ is the inertia-weighted task Jacobian pseudoinverse, given in the full-rank case by the following expression

$$\mathbf{J}_{t,H}^\dagger = \mathbf{H}^{-1} \mathbf{J}_t^\top (\mathbf{J}_t \mathbf{H}^{-1} \mathbf{J}_t^\top)^{-1} .$$

The solution (11.47) minimizes

$$\int_{t_i}^{t_f} \frac{1}{2} \dot{\mathbf{q}}^\top \mathbf{H}(\mathbf{q}) \dot{\mathbf{q}} dt ,$$

i. e., the integral over $[t_i, t_f]$ of the manipulator kinetic energy [11.52]. Once again, the correct boundary conditions must be used; for example, the case of free endpoints leads to

$$\dot{\mathbf{q}}(\bar{t}) = \mathbf{J}_{t,H}^\dagger \dot{\mathbf{t}}(\bar{t}) \quad \bar{t} = t_i, t_f .$$

A mathematical condition for cyclicity exists for a particular class of redundancy resolution methods [11.55]. In particular, consider any scheme of the form

$$\dot{\mathbf{q}} = \mathbf{G}_t(\mathbf{q}) \dot{\mathbf{t}} , \quad (11.48)$$

where \mathbf{G}_t is any *generalized inverse* of the task Jacobian \mathbf{J}_t , i. e., an $N \times M$ matrix such that $\mathbf{J}_t \mathbf{G}_t \mathbf{J}_t^\top = \mathbf{J}_t$ (the pseudoinverse matrix \mathbf{J}_t^\dagger in (11.13) is a particular

generalized inverse). Assume that the assigned task $\mathbf{t}(t)$ describes a cyclic trajectory in a simply connected region of the task space, and denote by $\mathbf{g}_{ti}(\mathbf{q})$ the i -th column of \mathbf{G}_t . A necessary and sufficient condition for (11.48) to generate cyclic joint trajectories is that the *distribution*

$$\Delta_{G_t}(\mathbf{q}) = \text{span}\{\mathbf{g}_{t1}(\mathbf{q}), \dots, \mathbf{g}_{tM}(\mathbf{q})\}$$

is involutive (i.e., it is closed under the Lie bracket operation).

It should be emphasized that the involutivity of Δ_{G_t} is a strong condition, because it must be satisfied at any configuration. This suggests that most generalized inverses are not cyclic. Note also that the above condition

for cyclicity depends on the chosen generalized inverse (i.e., pseudoinverse, weighted pseudoinverse, and so on) as well as on the form of \mathbf{J}_t , which in turn is related to the mechanical structure of the manipulator. This means that cyclicity must be established on a case-by-case basis.

As for redundancy resolution schemes which do not fall in the class (11.48) – namely, those entailed by the general inverse solution (11.12) – they are in general not cyclic. In particular, this is true for when local optimization is used to solve redundancy, as in (11.32). A notable exception is the extended Jacobian method, which is always cyclic.

11.8 Hyperredundant Manipulators

A natural question arising from consideration of kinematic redundancy is: what happens as the number of joints (and hence degree of redundancy) becomes much larger than the task-space dimension, and ultimately increases towards infinity? Examples of serial rigid-link systems with many joints exist in nature (snakes, and mammalian and fish backbones, for example), and these biological *existence proofs* have provided motivation and insight for robot analysts and hardware designers for many years. This interest has resulted in the development of several special sub-classes of redundant manipulators, collectively known as *hyperredundant manipulators*. This section reviews the underlying issues and state of the art in this interesting area.

A manipulator is generally considered to be hyperredundant if its controllable configuration (joint) space degrees of freedom are comparable to, or exceed, its task space degrees of freedom. (Hence, seven- or eight-DOF spatial rigid-link manipulators are not usually considered to be hyperredundant.) Hyperredundant manipulators have enhanced potential to use their extra joints for whole arm grasping/manipulation and maneuvering within tight obstacle fields. Their anticipated applications therefore include operations in congested environments (disaster relief, medical applications, etc.).

Since as we have seen the addition of (any) redundant degrees of freedom adds fundamentally new capabilities (self-motion, subtask performance capability), intuitively it would seem that the addition of abundant degrees of freedom would enable (and demand from motion planners) increasingly novel and sophisticated behaviors. However, on closer observation, many of the motivating biological hyperredundant systems actu-

ally perform relatively restricted movements, controlled by very simple algorithms. For example, (biological) snake backbones have very many joints, yet snake movement is typically restricted to approximately sinusoidal motions, governed by simple parameters (amplitude, frequency). Therefore there are some key simplifications which designers and algorithm developers have sought and made use of in the development of hyperredundant manipulators, as discussed in Sect. 11.8.3.

In terms of hardware development, the field has evolved in two major directions: *vertebrate-like* rigid-link designs, and *invertebrate-like* continuum manipulators. Summaries of the history and state of the art of each of these areas are given in the following two subsections. This is followed by a discussion of the unique analysis techniques required of and developed for hyperredundant robot systems. In all cases, the key issue is that of handling complexity, in one form or another.

11.8.1 Rigid-Link Hyperredundant Designs

This category of hyperredundant manipulators is perhaps the most *natural*, in terms of being the logical evolution from traditional serial rigid link manipulators (redundant or not). Designs in this category feature a hyperabundance of joints in their (usually serial) rigid-link structure. A good early example of this class of manipulator hardware is the JPL Serpentine Robot [11.56], which featured 12 degrees of freedom in a modular design.

The general trend has to shrink the size of the links, so that the robot structure resembles that of a biolog-



Fig. 11.4 17-DOF robot (courtesy Robotics Research Corporation)

ical *backbone*. This approach conceptually allows the development of manipulators with a very high degree of redundancy; for example, the 30-DOF planar manipulator developed at Caltech [11.57] remains perhaps the best known hyperredundant manipulator.

Note that various types of *nontraditional* rigid-link based manipulator systems can be classified as hyperredundant systems within this category. A novel elephant's trunk was based on a design featuring a 32-DOF backbone [11.58]. Several *dual-arm/torso* designs [11.59] including NASA's *special-purpose dextrous manipulator (SPDM)* [11.60] can be grouped in this category of hyperredundant manipulators, and researchers in the area of humanoid robots have made contributions to the area, both in terms of theory and hardware.

Another important special group in this category is that of robotic snakes. This area owes much to the pioneering work of Hirose [11.61], who analyzed the movements of (biological) snakes and incorporated the understanding generated into the design and control of a series of innovative snake-like robots. Of particular in-



Fig. 11.5 The NASA special-purpose dextrous manipulator (SPDM) (courtesy MDA Federal)



Fig. 11.6 Active cord mechanism (courtesy S. Hirose)

terest here were the series of hyperredundant *active cord mechanism (ACM)* robots developed by Hirose's group.

Numerous other snake robot designs have been proposed over the years, and the area continues to be highly active.

The key conceptual advantage of rigid-link hyperredundant designs is that they inherit the kinematics of their conventional industrial manipulator counterparts. Therefore, all the traditional modeling techniques (for example Denavit–Hartenberg-based approaches for modeling the forward kinematics and finding the manipulator Jacobian) apply. Therefore, in theory all the techniques discussed earlier in this chapter apply directly to this category of hyperredundant manipu-



Fig. 11.7 The OctArm continuum manipulator (courtesy I. Walker)

lator (although complexity of the resulting models is a key obstacle to analysis and understanding). However, the second main category, overviewed in the following subsection, by taking hyperredundancy to its logical extreme, does not inherit these particular advantages.

11.8.2 Continuum Robot Designs

The term *continuum manipulator*, first coined in [11.62], indicates the hyperredundant robot concept taken to the extreme. In other words, conceptually this category of robots have backbone structures *taken to the limit*, with their number of *joints* tending to infinity, but with their *link lengths* tending to zero. While this concept initially appears idealistic, complicated, and impossible to implement, the limiting case (a smooth *continuum* curve, with the ability to bend at any point along the backbone) is quite easy to realize in hardware. Numerous continuum designs have been proposed since the early 1960s [11.62, 63]. The key design issues revolve around how to actuate (bend and possibly extend/contract) the backbone. Two basic design strategies have emerged: (1) extrinsic actuation, where the actuators are separate from the backbone structure [11.64]; and (2) intrinsic actuation, where the actuators represent a fundamental part of the backbone, e.g., [11.65]. Tendons have proved a popular and generally successful choice for intrinsically actuated continuum designs [11.58, 64]. Artificial muscle technologies, notably *McKibben* muscles, have proved effective in intrinsically actuated hardware realizations [11.65].

A commercially successful group of extrinsically actuated continuum manipulators are currently manu-



Fig. 11.8 The continuum manipulator (courtesy OCR Robotics)

factured and retailed by OCR Robotics in the United Kingdom [11.64].

While continuum manipulators are thus seen to be the extreme case of hyperredundancy, it was quickly realized that they represent a fundamentally new class of manipulators [11.62]. A continuous flexible backbone features, at least in theory, an infinite number of degrees of freedom. As such, the traditional tools in robotics for modeling (a finite number of) serial rigid links no longer apply. Additionally, it is obviously not possible to actuate an infinite number of degrees of freedom in practice. Continuum manipulator hardware universally features a finite number of actuators, applying forces/torques to the backbone at a fixed and preselected set of locations. Continuum robots are therefore inherently both hyperredundant and underactuated. This causes significant complexity in their analysis. However, significant progress has been made, and understanding resulting from analysis of continuum kinematics has been applied to the benefit of both continuum and non-continuum hyperredundant manipulators, as discussed in the following subsection.

11.8.3 Hyperredundant Manipulator Modeling

Clearly, to correctly model continuum manipulators, models based on continuous backbones are required. Interestingly, such models have also proved to be key theoretical resources in the motion planning of rigid-link hyperredundant systems. As noted earlier, in theory discrete-link hyperredundant manipulators can be modeled using (and their planning based on) traditional manipulator modeling techniques. However, it was quickly found that the computational complexity of such approaches made them difficult to implement, and the corresponding models hard to visualize. An alternative strategy (and a more successful one in practice) has been to adopt an approach based on continuum manipulator kinematics. In the following, we overview the recent development of continuum manipulator kinematics. We then discuss how the existence of both discrete-link and continuum models have enhanced motion planning for each category of hyperredundant system.

The lack of distinct links in continuum systems makes the standard robot manipulator modeling strategy of a finite number of coordinate frames (each fixed in one link) inappropriate for their modeling. Instead, the natural approach is to model continuum kinematics via a frame that evolves along the backbone, parameterized by arc length s . Local motion of the backbone at point s is modeled in terms of the local frame. This strategy allows computation of *forward kinematics*, and the construction of *continuum Jacobians*, analogous to those for rigid-link systems [11.65, 66].

Numerous alternatives for selection of the backbone frame have been proposed [11.67, 68]. A popular choice is the well-known Serret–Frenet frame [11.66], which evolves along the backbone according to:

$$\begin{aligned}\frac{d\mathbf{t}}{ds} &= \kappa \mathbf{n} , \\ \frac{d\mathbf{n}}{ds} &= -\kappa \mathbf{t} + \tau \mathbf{b} , \\ \frac{d\mathbf{b}}{ds} &= -\tau \mathbf{n} .\end{aligned}$$

In the above, the frame origin is given by \mathbf{x} , where the unit tangent to the curve $\mathbf{t} = d\mathbf{x}/ds$, and forms one of the frame axes. The other axes are defined as the normal ($\mathbf{t} \cdot \mathbf{n} = 0$) and the binormal $\mathbf{b} = \mathbf{t} \times \mathbf{n}$. The curvature κ and torsion τ dictate the shape of the curve. The axes of the Serret–Frenet frame provide an intuitive visualization of local movement: two possible dimensions of bending, corresponding to rotations

about the normal and binormal axes, and one of extension/contraction (present and controllable in several continuum robots), corresponding to translation along the tangent axis. The Frenet–Serret model has the advantage of being a well-established means for modeling continuous spatial curves. Alternative frame representations in the continuum robot literature have selected the axes of the frame to align with the controlled motion axes of particular hardware realizations [11.68]. In either case, given a frame, the key problem is how to use the resulting model to plan motions for hyperredundant robots. The basic issue is that the continuum kinematics models in their basic form feature infinite degrees of freedom (as necessary to model arbitrary spatial curves). However, hyperredundant robots (discrete-joint or continuum) can be controlled in only a finite number of ways, thus admitting a reduced set of physical solutions. Therefore, research in hyperredundant manipulators has concentrated on how to constrain continuum kinematics models to best represent robot hardware.

The initial major breakthrough in motion planning for hyperredundant arms was made in a landmark series of papers [11.57, 67, 69] introducing continuum kinematics and using them to approximate rigid-link hyperredundant systems [11.67]. Basically the philosophy is to use a (theoretical) curve to model the physical backbone of a hyperredundant robot. Motion planning is performed for the curve, and the (discrete) robot backbone is then *fitted* to the resulting (continuous) solution curve. This approach has proved quite effective, and the associated research introduced several key theoretical concepts to the area. In particular, the use of a modal approach [11.69] (restricting the classes of allowable solutions to shapes generated via simple linear combinations of *modes*), arose from this approach to redundancy resolution for hyperredundant manipulators. This influential concept can be viewed as a *top-down* approach of building a general model, and adapting it (via mode selection and curve fitting) to hardware.

More recent research has concentrated on the *dual* notion to that of the previous section, using the physical constraints imposed on the backbones of specific continuum robot hardware types to construct continuum kinematics. This can be viewed as a *bottom-up* approach, focused on particular hardware classes, with the main aim of sufficiently modeling the hardware to avoid the use of approximations in the motion planning. Examples of this approach are given in [11.65, 68], where the key constraint imposed on a general continuum kinematics model arises from the observation that many continuum hardware implementations resolve into

a finite number of constant curvature sections. (This arises naturally from the imposition of a finite number of input force/torques on a stiff continuum backbone.) In [11.65] it is noted that the resulting constant-curvature models can be viewed as a particular case of the general modal approach of [11.69].

The above approach [11.65] involves several transformations, to transition the map between task and actuator space via a *section* space. As a key part of this transformation, a conventional (theoretical) rigid-link model is used to model the kinematics of each section. Thus rigid-link kinematics have proven key to resolving redundancy for continuum manipulators, as continuum kinematics play a key part in redundancy

resolution for rigid-link hyperredundant arms. Finally, a series of Jacobians are formed. The difference here is that the continuum Jacobian is a function of locally meaningful variables (bending angle, curvature, and extension) defining section shape or the direct values of the actuators that determine these variables. Given the existence of continuum Jacobians, the redundancy is usually resolved as

$$\dot{\mathbf{i}} = \mathbf{J}_E^\dagger \dot{\mathbf{i}}_E + (\mathbf{I} - \mathbf{J}_E^\dagger \mathbf{J}_E) \dot{\mathbf{i}}_0$$

in the same way as for conventional redundant manipulators (and with the same general corresponding advantages and issues), as discussed earlier in the chapter.

11.9 Conclusion and Further Reading

Research on kinematically redundant robots has been flourishing for over two decades now, and is still very active. The number of papers dealing with this subject is therefore enormous. The few works cited below are simply a small addition to the fundamental contributions already referenced so far, and constitute by no means an exhaustive list.

The mechanical design of kinematically redundant manipulators has been studied in many papers, see, e.g., [11.70–74]. In particular, the superiority of human-arm-like manipulators over conventional 6-DOF robots was first advocated in [11.75]. Reconfiguration of this arm so as to ensure full mobility in the whole workspace is studied in [11.76, 77].

A general analysis of the inverse kinematic problem for redundant manipulators is presented in [11.78]. In particular, the geometric structure of self-motions is analyzed in [11.79].

Weighting the damped least-squares solution for guaranteed singularity avoidance in anisotropic end-effector tasks was proposed in [11.9, 21, 80].

In addition to singularity avoidance, redundancy has also been exploited to achieve obstacle avoidance [11.37, 46, 81, 82], minimization of the effects of joint elasticity [11.83], fault tolerance [11.84], reduction of impact force [11.85, 86], and maximization of various dexterity measures [11.4, 5, 43, 87]. A completely different approach for obstacle avoidance with redundant robots is presented in [11.88].

A review of redundancy resolution via local optimization is given in [11.89]. A numerically efficient alternative for redundancy resolution via local optimization is proposed in [11.90]. A redundancy resolution approach with somehow intermediate characteristics between local and global optimization is presented in [11.91]. Other methods for second-order redundancy resolution are discussed, e.g., in [11.92]. Also worth citing is the dynamically consistent generalized inverse of [11.93].

The cyclicity issue was first pointed out in [11.94], and further investigated, e.g., in [11.95, 96]. General formalisms for redundancy resolution of vehicle-manipulator systems subject to nonholonomic constraints are described in [11.97, 98].

References

- 11.1 D.E. Whitney: Resolved motion rate control of manipulators and human prostheses, *IEEE Trans. Man-Mach. Syst.* **10**(2), 47–53 (1969)
- 11.2 G.H. Golub, C. Reinsch: Singular value decomposition and least-squares solutions, *Numer. Math.* **14**, 403–420 (1970)
- 11.3 A.A. Maciejewski, C.A. Klein: The singular value decomposition: computation and applications to robotics, *Int. J. Robot. Res.* **8**(6), 63–79 (1989)
- 11.4 T. Yoshikawa: Manipulability of robotic mechanisms, *Int. J. Robot. Res.* **4**(2), 3–9 (1985)

- 11.5 C.A. Klein, B.E. Blaho: Dexterity measures for the design and control of kinematically redundant manipulators, *Int. J. Robot. Res.* **6**(2), 72–83 (1987)
- 11.6 A.A. Maciejewski, C.A. Klein: Numerical filtering for the operation of robotic manipulators through kinematically singular configurations, *J. Robot. Syst.* **5**, 527–552 (1988)
- 11.7 S. Chiaverini: Estimate of the two smallest singular values of the Jacobian matrix: application to damped least-squares inverse kinematics, *J. Robot. Syst.* **10**, 991–1008 (1993)
- 11.8 O. Egeland, M. Ebdstrup, S. Chiaverini: Sensory control in singular configurations – application to visual servoing, *IEEE Int. Workshop Intell. Motion Contr.* (Istanbul 1990) pp. 401–405
- 11.9 Y. Nakamura, H. Hanafusa: Inverse kinematic solutions with singularity robustness for robot manipulator control, *Trans. ASME – J. Dyn. Syst. Meas. Contr.* **108**, 163–171 (1986)
- 11.10 D.E. Whitney: The mathematics of coordinated control of prosthetic arms and manipulators, *Trans. ASME – J. Dyn. Syst. Meas. Contr.* **94**, 303–309 (1972)
- 11.11 T.L. Boullion, P.L. Odell: *Generalized Inverse Matrices* (Wiley-Interscience, New York 1971)
- 11.12 C.R. Rao, S.K. Mitra: *Generalized Inverse of Matrices and its Applications* (Wiley, New York 1971)
- 11.13 A. Ben-Israel, T.N.E. Greville: *Generalized Inverses: Theory and Applications* (Wiley, New York 1974)
- 11.14 R.H. Taylor: Planning and execution of straight-line manipulator trajectories, *IBM J. Res. Dev.* **23**, 424–436 (1979)
- 11.15 M. Sampei, K. Furuta: Robot control in the neighborhood of singular points, *IEEE J. Robot. Autom.* **4**, 303–309 (1988)
- 11.16 E.W. Aboaf, R.P. Paul: Living with the singularity of robot wrists, *IEEE Int. Conf. Robot. Autom.* (Raleigh 1987) pp. 1713–1717
- 11.17 S. Chiaverini, O. Egeland: A solution to the singularity problem for six-joint manipulators, *IEEE Int. Conf. Robot. Autom.* (Cincinnati 1990) pp. 644–649
- 11.18 S. Chiaverini, O. Egeland: An efficient pseudo-inverse solution to the inverse kinematic problem for six-joint manipulators, *Model. Identif. Contr.* **11**(4), 201–222 (1990)
- 11.19 O. Khatib: A unified approach for motion and force control of robot manipulators: the operational space formulation, *IEEE J. Robot. Autom.* **3**, 43–53 (1987)
- 11.20 C.W. Wampler II: Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods, *IEEE Trans. Syst. Man Cybern.* **16**, 93–101 (1986)
- 11.21 S. Chiaverini, O. Egeland, R.K. Kanestrøm: Achieving user-defined accuracy with damped least-squares inverse kinematics, *5th Int. Conf. Adv. Robot.* ('91 ICAR) (Pisa 1991) pp. 672–677
- 11.22 J.R. Sagli: Coordination of Motion in Manipulators with Redundant Degrees of Freedom. Ph.D. Thesis (Institut for Teknisk Kibernetikk, Norges Tekniske Høgskole, Trondheim, N 1991)
- 11.23 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy, *Int. J. Robot. Res.* **10**(4), 410–425 (1991)
- 11.24 B. Siciliano: A closed-loop inverse kinematic scheme for on-line joint-based robot control, *Robotica* **8**, 231–243 (1990)
- 11.25 Z.R. Novaković, B. Siciliano: A new second-order inverse kinematics solution for redundant manipulators. In: *Advances in Robot Kinematics*, ed. by S. Stifter, J. Lenarčič (Springer-Verlag, Vienna, Austria 1991) pp. 408–415
- 11.26 A. Balestrino, G. De Maria, L. Sciavicco: Robust control of robotic manipulators, *9th IFAC World Congress* (Budapest 1984) pp. 80–85
- 11.27 W.A. Wolovich, H. Elliott: A computational technique for inverse kinematics, *23rd IEEE Conf. Decis. Contr.* (Las Vegas 1984) pp. 1359–1363
- 11.28 L. Sciavicco, B. Siciliano: A solution algorithm to the inverse kinematic problem for redundant manipulators, *IEEE J. Robot. Autom.* **4**, 403–410 (1988)
- 11.29 J. Baillieul, J. Hollerbach, R.W. Brockett: Programming and control of kinematically redundant manipulators, *23th IEEE Conf. Decis. Contr.* (Las Vegas 1984) pp. 768–774
- 11.30 A. Liégeois: Automatic supervisory control of the configuration and behavior of multibody mechanisms, *IEEE Trans. Syst. Man Cybern.* **7**, 868–871 (1977)
- 11.31 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, *IEEE Int. Conf. Robot. Autom.* (St. Louis 1985) pp. 500–505
- 11.32 J.C. Latombe: *Robot Motion Planning* (Kluwer Academic, Boston 1991)
- 11.33 D.G. Luenberger: *Linear and Nonlinear Programming* (Addison-Wesley, Reading 1984)
- 11.34 A. De Luca, G. Oriolo: Issues in acceleration resolution of robot redundancy, *3rd IFAC Symp. Robot Contr.* (Vienna 1991) pp. 665–670
- 11.35 J. Baillieul: Kinematic programming alternatives for redundant manipulators, *IEEE Int. Conf. Robot. Autom.* (St. Louis 1985) pp. 722–728
- 11.36 P.H. Chang: A closed-form solution for inverse kinematics of robot manipulators with redundancy, *IEEE J. Robot. Autom.* **3**, 393–403 (1987)
- 11.37 J. Baillieul: Avoiding obstacles and resolving kinematic redundancy, *IEEE Int. Conf. Robot. Autom.* (San Francisco 1986) pp. 1698–1704
- 11.38 L. Sciavicco, B. Siciliano: Solving the inverse kinematic problem for robotic manipulators, *6th CISM-IFTOMM Symp. Theory Practice Robots Manip.* (Kraków 1986) pp. 107–114
- 11.39 L. Sciavicco, B. Siciliano: A dynamic solution to the inverse kinematic problem for redundant manipu-

- lators, IEEE Int. Conf. Robot. Autom. (Raleigh 1987) pp.1081–1087
- 11.40 O. Egeland: Task-space tracking with redundant manipulators, IEEE J. Robot. Autom. **3**, 471–475 (1987)
- 11.41 H. Seraji: Configuration control of redundant manipulators: theory and implementation, IEEE J. Robot. Autom. **5**, 472–490 (1989)
- 11.42 J. Baillieul: A constraint oriented approach to inverse problems for kinematically redundant manipulators, IEEE Int. Conf. Robot. Autom. (Raleigh 1987) pp.1827–1833
- 11.43 S.L. Chiu: Control of redundant manipulators for task compatibility, IEEE Int. Conf. Robot. Autom. (Raleigh 1987) pp.1718–1724
- 11.44 H. Seraji, R. Colbaugh: Improved configuration control for redundant robots, J. Robot. Syst. **7**, 897–928 (1990)
- 11.45 O. Egeland, J.R. Sagli, I. Spangelo, S. Chiaverini: A damped least-squares solution to redundancy resolution, IEEE Int. Conf. Robot. Autom. (Sacramento 1991) pp. 945–950
- 11.46 A.A. Maciejewski, C.A. Klein: Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments, Int. J. Robot. Res. **4**(3), 109–117 (1985)
- 11.47 Y. Nakamura, H. Hanafusa, T. Yoshikawa: Task-priority based redundancy control of robot manipulators, Int. J. Robot. Res. **6**(2), 3–15 (1987)
- 11.48 H. Hanafusa, T. Yoshikawa, Y. Nakamura: Analysis and control of articulated robot arms with redundancy, IFAC 8th Triennial World Congress (Kyoto 1981) pp.78–83
- 11.49 Y. Nakamura, H. Hanafusa: Task priority based redundancy control of robot manipulators. In: *Robotics Research – The Second International Symposium*, ed. by H. Hanafusa, H. Hinoue (MIT Press, Cambridge 1985) pp.155–162
- 11.50 S. Chiaverini: Task-priority redundancy resolution with robustness to algorithmic singularities, 4th IFAC Symp. Robot Contr. (Capri 1994) pp.393–399
- 11.51 S. Chiaverini: Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, IEEE Trans. Robot. Autom. **13**, 398–410 (1997)
- 11.52 K. Kazerounian, Z. Wang: Global versus local optimization in redundancy resolution of robotic manipulators, Int. J. Robot. Res. **7**(5), 312 (1988)
- 11.53 J.M. Hollerbach, K.C. Suh: Redundancy resolution of manipulators through torque optimization, IEEE J. Robot. Autom. **3**, 308–316 (1987)
- 11.54 J.M. Hollerbach, K.C. Suh: Local versus global torque optimization of redundant manipulators, IEEE Int. Conf. Robot. Autom. (Raleigh 1987) pp. 619–624
- 11.55 T. Shamir, Y. Yomdin: Repeatability of redundant manipulators: Mathematical solution of the problem, IEEE Trans. Autom. Contr. **33**, 1004–1009 (1988)
- 11.56 E. Paljug, T. Ohm, S. Hayati: The JPL serpentine robot: a 12-DoF system for inspection, IEEE Int. Conf. Robot. Autom. (1995) pp.3143–3148
- 11.57 G.S. Chirikjian, J.W. Burdick: Design and experiments with a 30 DoF robot, IEEE Int. Conf. Robot. Autom. (1993) pp.113–117
- 11.58 M.W. Hannan, I.D. Walker: Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots, J. Robot. Syst. **20**(2), 45–63 (2003)
- 11.59 J.P. Karlen, J.M. Thompson, H.I. Vold, J.D. Farrell, P.H. Eismann: A dual-arm dexterous manipulator system with anthropomorphic kinematics, IEEE Int. Conf. Robot. Autom. (1990) pp.368–373
- 11.60 G. Hirzinger, B. Brunner, R. Lampariello, J. Schott, B.M. Steinmetz: Advances in orbital robotics, IEEE Int. Conf. Robot. Autom. (2000) pp. 898–907
- 11.61 S. Hirose: *Biologically inspired robots* (Oxford Univ. Press, Oxford 1993)
- 11.62 G. Robinson, J.B.C. Davies: Continuum robots – a state of the art, IEEE Int. Conf. Robot. Autom. (Detroit 1999) pp.2849–2854
- 11.63 V.C. Anderson, R.C. Horn: Tensor arm manipulator design, Trans. ASME **67-DE-57**, 1–12 (1967)
- 11.64 R. Buckingham: Snake arm robots, Ind. Robot Int. J. **29**(3), 242–245 (2002)
- 11.65 B.A. Jones, I.D. Walker: Kinematics for multisection continuum robots, IEEE Trans. Robot. **22**(1), 43–55 (2006)
- 11.66 H. Mochiyama, E. Shimemura, H. Kobayashi: Shape correspondence between a spatial curve and a manipulator with hyper degrees of freedom, IEEE Int. Conf. Robot. Autom. (1998) pp.161–166
- 11.67 G.S. Chirikjian: Theory and Applications of Hyperredundant Robotic Mechanisms. Ph.D. Thesis (Department of Applied Mechanics, California Institute of Technology 1992)
- 11.68 I.A. Gravagne, I.D. Walker: Manipulability, force, and compliance analysis for planar continuum manipulators, IEEE Trans. Robot. Autom. **18**(3), 263–273 (2002)
- 11.69 G.S. Chirikjian, J.W. Burdick: A modal approach to hyper-redundant manipulator kinematics, IEEE Trans. Robot. Autom. **10**(3), 343–354 (1994)
- 11.70 J. Salisbury, J. Abramowitz: Design and control of a redundant mechanism for small motion, IEEE Int. Conf. Robot. Autom. (St. Louis 1985) pp.323–328
- 11.71 J. Baillieul: Design of kinematically redundant mechanisms, 24th IEEE Conf. Decis. Contr. (Ft. Lauderdale 1985) pp.18–21
- 11.72 G.S. Chirikjian, J.W. Burdick: Design and experiments with a 30 DoF robot, IEEE Int. Conf. Robot. Autom. (Atlanta 1993) pp.113–119
- 11.73 J. Angeles: The design of isotropic manipulator architectures in the presence of redundancies, Int. J. Robot. Res. **11**(3), 196–201 (1992)
- 11.74 A. Bowling, O. Khatib: Design of macro/mini manipulators for optimal dynamic performance,

- IEEE Int. Conf. Robot. Autom. (Albuquerque 1997) pp. 449–454
- 11.75 J.M. Hollerbach: Optimum kinematic design for a seven degree of freedom manipulator. In: *Robotics Research – The Second International Symposium*, ed. by H. Hanafusa, H. Hinoue (MIT Press, Cambridge 1985) pp. 216–222
- 11.76 O. Egeland, J.R. Sagli, S. Hendseth, F. Wilhelmsen: Dynamic coordination in a manipulator with seven joints, IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 125–130
- 11.77 S. Chiaverini, B. Siciliano, O. Egeland: Kinematic analysis and singularity avoidance for a seven-joint manipulator, Am. Contr. Conf. (San Diego 1990) pp. 2300–2305
- 11.78 D.R. Baker, C.W. Wampler: On the inverse kinematics of redundant manipulators, Int. J. Robot. Res. 7(2), 3–21 (1988)
- 11.79 J.W. Burdick: On the inverse kinematics of redundant manipulators: characterization of the self-motion manifolds, IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 264–270
- 11.80 S. Chiaverini, O. Egeland, R.K. Kanestrøm: Weighted damped least-squares in kinematic control of robotic manipulators, Adv. Robot. 7, 201–218 (1993)
- 11.81 M. Kirčanski, M. Vukobratović: Trajectory planning for redundant manipulators in presence of obstacles, 5th CISM-IFTOMM Symp. Theors Pract. Robots Manip. (Udine 1984) pp. 43–58
- 11.82 C.A. Klein: Use of redundancy in the design of robotic systems. In: *Robotics Research – The Second International Symposium*, ed. by H. Hanafusa, H. Hinoue (MIT Press, Cambridge 1985) pp. 207–214
- 11.83 J. Baillieul: Kinematic redundancy and the control of robots with flexible components, IEEE Int. Conf. Robot. Autom. (Nice 1992) pp. 715–721
- 11.84 J.D. English, A.A. Maciejewski: Fault tolerance for kinematically redundant manipulators: anticipating free-swinging joint failures, 14, 566–575 (1998)
- 11.85 I.D. Walker: The use of kinematic redundancy in reducing impact and contact effects in manipulation, IEEE Int. Conf. Robot. Autom. (Cincinnati 1990) pp. 434–439
- 11.86 M.W. Gertz, J.O. Kim, P.K. Khosla: Exploiting redundancy to reduce impact force, IEEE/RSJ Int. Workshop Intell. Robot. Syst. (Osaka 1991) pp. 179–184
- 11.87 T. Yoshikawa: Dynamic manipulability of robot manipulators, J. Robot. Syst. 2(1), 113–124 (1985)
- 11.88 G. Oriolo, M. Ottavi, M. Vendittelli: Probabilistic motion planning for redundant robots along given end-effector paths, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Lausanne 2002) pp. 1657–1662
- 11.89 D.N. Nenchev: Redundancy resolution through local optimization: a review, J. Robot. Syst. 6, 6 (1989)
- 11.90 A. De Luca, G. Oriolo: The reduced gradient method for solving redundancy in robot arms, Robotersysteme 7(2), 117–122 (1991)
- 11.91 S. Seereeram, J.T. Wen: A global approach to path planning for redundant manipulators, IEEE Trans. Robot. Autom. 11(1), 152–160 (1995)
- 11.92 A. De Luca, G. Oriolo, B. Siciliano: Robot redundancy resolution at the acceleration level, Lab. Robot. Autom. 4(2), 97–106 (1992)
- 11.93 O. Khatib: Inertial properties in robotics manipulation: an object-level framework, Int. J. Robot. Res. 14(1), 19–36 (1995)
- 11.94 C.A. Klein, C.H. Huang: Review of pseudoinverse control for use with kinematically redundant manipulators, IEEE Trans. Syst. Man Cybern. 13, 245–250 (1983)
- 11.95 R. Mukherjee: Design of holonomic loops for repeatability in redundant manipulators, IEEE Int. Conf. Robot. Autom. (Nagoya 1985) pp. 2785–2790
- 11.96 A. De Luca, G. Oriolo: Nonholonomic behavior in redundant robots under kinematic control, IEEE Trans. Robot. Autom. 13(5), 776–782 (1997)
- 11.97 B. Bayle, J.Y. Fourquet, M. Renaud: Manipulability of wheeled-mobile manipulators: application to motion generation, Int. J. Robot. Res. 22(7–8), 565–581 (2003)
- 11.98 A. De Luca, G. Oriolo, P.R. Giordano: Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators, IEEE Int. Conf. Robot. Autom. (Orlando 2006) pp. 1867–1873

Parallel Mech

12. Parallel Mechanisms and Robots

Jean-Pierre Merlet, Clément Gosselin

This chapter presents an introduction to the kinematics and dynamics of parallel mechanisms, also referred to as parallel robots. As opposed to classical serial manipulators, the kinematic architecture of parallel robots includes *closed-loop kinematic chains*. As a consequence, their analysis differs considerably from that of their serial counterparts. This chapter aims to presenting the fundamental formulations and techniques used in their analysis.

12.1	Definitions	269
12.2	Type Synthesis of Parallel Mechanisms ...	271
12.3	Kinematics	271
12.3.1	Inverse Kinematics	271
12.3.2	Forward Kinematics	272
12.4	Velocity and Accuracy Analysis	273
12.5	Singularity Analysis	274
12.5.1	General Formulation	274
12.5.2	Parallel Singularity Analysis	275
12.6	Workspace Analysis	276
12.7	Static Analysis and Static Balancing	277
12.8	Dynamic Analysis	279
12.9	Design	279
12.10	Application Examples	280
12.11	Conclusion and Further Reading	281
	References	281

12.1 Definitions

A *closed-loop kinematic chain* is one in which the links and joints are arranged such that at least one closed loop exists. Furthermore, a *complex closed-loop kinematic chain* is obtained when one of the links – other than the base – has a degree of connectivity greater than or equal to three, i. e., one link other than the base is connected through joints to at least three other links. A *parallel manipulator* can be defined as a closed-loop mechanism composed of an end-effector having n degrees of freedom and a fixed base, linked together by at least two independent kinematic chains.

An example of such a structure was patented in 1928 by Gwinnett [12.1] to be used as a platform for a movie theater. In 1947 Gough [12.2] established the basic principles of a mechanism with a closed-loop kinematic structure (Fig. 12.1) that allows the positioning and orientation of a moving platform so as to test tire wear and tear. He built a prototype of this machine in 1955. For this mechanism, the moving effector is a hexagonal

platform whose vertices are all connected to a link by a ball-and-socket (spherical) joint. The other end of the link is attached to the base by a universal joint. A linear actuator allows the modification of the total length of the link; this mechanism is therefore a closed-loop kinematic structure, actuated by six linear actuators.

In 1965, Stewart [12.3] suggested the use of such a structure for flight simulators and the Gough mechanism is sometimes referred to as a *Stewart platform*. The same architecture was also concurrently proposed by Kappel as a motion simulation mechanism [12.4]. Nowadays the Gough platform is the platform of choice for flight simulators. The latter application provides a convincing illustration of the main advantage of parallel robots namely, their load-carrying capacity. Indeed, while the ratio of the mass of the payload over the mass of the robot is typically smaller than 0.15 for serial 6R industrial robots, this ratio can be larger than 10 for parallel structures. Another advantage of the Gough platform is

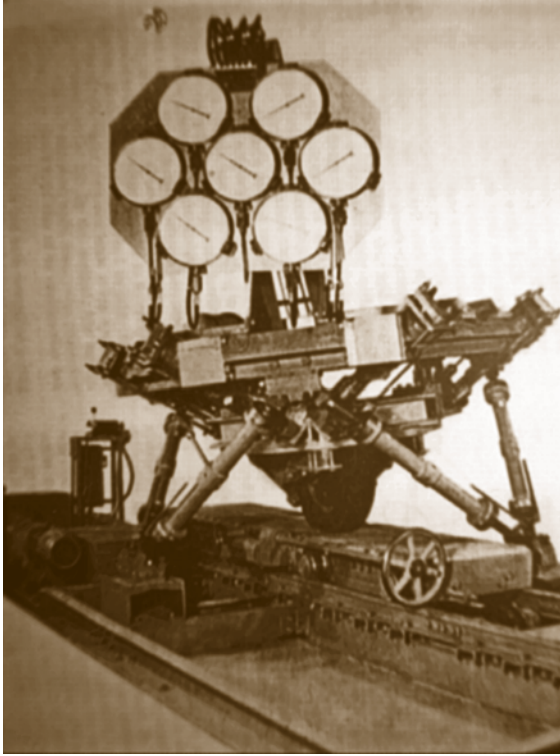


Fig. 12.1 Gough platform (1947)

its very good positioning accuracy. This high accuracy arises from the fact that the legs are working essentially in tension/compression and are subjected to virtually no bending – thereby leading to very small deformations – and from the fact that the errors in the internal sensors of the robot (the measurement of the lengths of the legs for the Gough platform) are mapped into very small errors of the platform position. Parallel robots are also almost insensitive to scaling (the same structure can be used for large or micro robots) and they can be built using almost any type of actuator or transmission, for example, wire transmissions can be used (see the *Robocrane* [12.5]). The main drawbacks of parallel robots are their small workspace and the singularities that can appear within the latter.

Apart from the Gough platform, the most successful designs of parallel robots are the Delta robot proposed by *Clavel* [12.6] (Fig. 12.2) and some planar parallel robots. The most common planar parallel robots have three identical legs each having an $R\underline{P}R$ or $\underline{R}RR$ architecture, where the underlined joint is the actuated one;

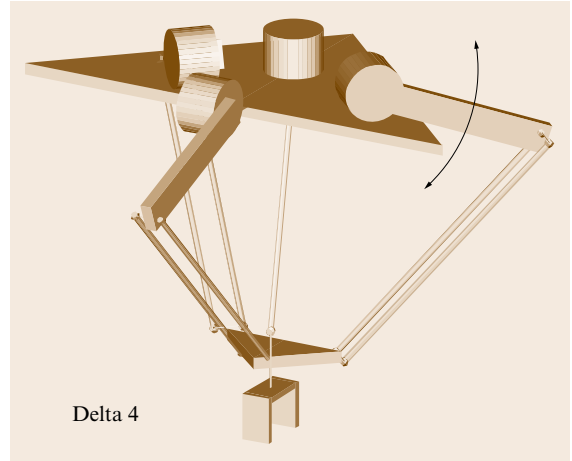


Fig. 12.2 The Delta robot

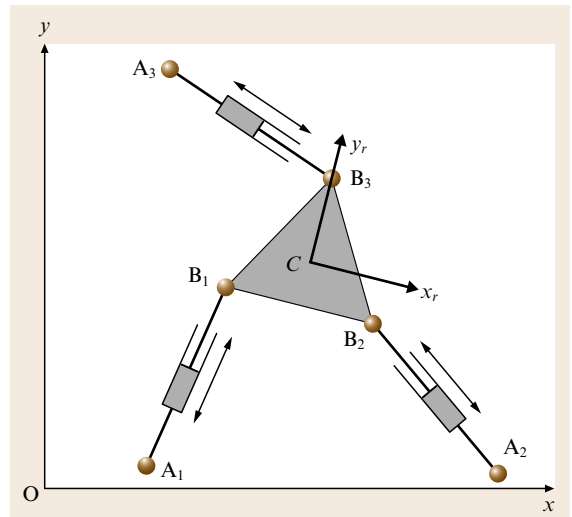


Fig. 12.3 The 3 – $R\underline{P}R$ planar parallel robot

such robots are often denoted as 3 – $R\underline{P}R$ (see Fig. 12.3) and 3 – $\underline{R}RR$.

The geometric arrangement of the joints and links of the Delta structure provides three translational degrees of freedom at the platform. Numerous other types of parallel robots have been proposed in recent years. Although most existing architectures are based on the intuition of their designer, the synthesis of parallel mechanisms can be dealt with systematically. An overview of the main approaches to the type synthesis of parallel mechanisms is given in the next section.

12.2 Type Synthesis of Parallel Mechanisms

Determining all potential mechanical architectures of parallel robots that can produce a given motion pattern at the moving platform is a challenging problem. Several authors have addressed this problem, referred to as type synthesis. The approaches proposed in the literature can be divided into three groups:

- *Approaches based on graph theory*: the enumeration of all possible structures having a given number of degrees of freedom (DOFs) can be performed by considering that there is only a finite set of possible kinematic pairs, and hence a very large, but finite, set of possible structure combinations (see, for instance, [12.7]). Classical mobility formulae – e.g., the Chebychev–Grübler–Kutzbach formula – are then used to determine the number of DOFs in the structure. Unfortunately these formulae do not take into account geometric properties of the mechanism that can change the number of DOFs of the platform. Therefore, synthesis approaches based strictly on graph theory can produce only limited results and have been largely superseded by the other two groups.
- *Approaches based on group theory*: the motion of a rigid body has the special structure of a group, the *displacement group*. Subgroups of the group of displacements, such as the spatial translations or all the translational and rotational motions about all axes that are parallel to the axis defined by a given vector (Schönflies motion) play an important role as they can be combined through the *intersection* operation [12.8] when elements of subgroups act on the same rigid body. Type synthesis consists in determining all the possible subgroups to which the different kinematic chains that will constitute the legs of the robot may belong so that their intersection leads to the desired motion pattern for the platform. Synthesis approaches based on group theory led to the discovery of numerous possible architectures. Nevertheless, the displacement group has special properties that are not reflected by its group struc-

ture alone. Also, the approach is limited to motion patterns that can be described by a subgroup of the displacement group.

- *Approaches based on screw theory*: in this approach, the first step is to determine the wrench system S that is reciprocal to the desired velocity twist of the moving platform. Then, the wrenches of the kinematic chains of the robot whose union spans the system S (and that determine all the possible structures of the kinematic chains that will generate the corresponding wrenches) are enumerated. Then, since all considered twists and wrenches are instantaneous, it is necessary to verify that the mobility of the platform is full-cycle and not only instantaneous. A systematic implementation of this approach is provided in [12.9].

These synthesis methods have been used to generate a large number of architectures that cannot be presented in this book, but the Web site of Merlet [12.10] presents a comprehensive description of a large number of mechanical architectures. Among others, several remarkable architectures were proposed in the recent years to fully or partially decouple translational motions or Schönflies motions (see [12.9] and the references therein).

It should also be pointed out that for robots with fewer than six DOFs, synthesis methods generally lead to architectures that must satisfy stringent geometric constraints (such as for the Delta robot, where the axes of the parallelograms should be parallel and their sides of equal length). Such constraints cannot be exactly verified in practice and consequently the manipulator will exhibit *parasitic*, i.e., unwanted, motion. Open issues therefore include:

- the determination of the maximal amplitude of the parasitic motion over a given workspace and for a given robot
- the determination of the geometry of a given structure such that the maximal amplitude of the parasitic motion will be smaller than a given threshold

12.3 Kinematics

12.3.1 Inverse Kinematics

The solution of the inverse kinematic problem is usually straightforward for parallel robots. This can be illus-

trated with the Gough platform. The solution of the inverse kinematic problem consists in determining the vector of leg lengths \mathbf{q} for a given pose of the platform defined by the position vector \mathbf{p} of a given point on the

platform in a fixed reference frame and a rotation matrix \mathbf{R} representing the orientation of the platform with respect to the fixed frame. Let \mathbf{a}_i denote the position vector of the anchor point of the i th leg on the base given in the fixed reference frame and \mathbf{b}_i denote the position vector of the anchor point of the i th leg on the platform, given in a moving frame attached to the platform. The length of the i th leg is the norm of the vector connecting the two anchor points. The latter vector, denoted by \mathbf{s}_i , can be written as

$$\mathbf{s}_i = \mathbf{p} + \mathbf{R}\mathbf{b}_i - \mathbf{a}_i, \quad i = 1, \dots, 6. \quad (12.1)$$

Given the pose of the platform (vector \mathbf{p} and matrix \mathbf{R}), the vector \mathbf{s}_i is readily calculated using (12.1), and hence the leg lengths can be obtained.

12.3.2 Forward Kinematics

The solution of the forward kinematic problem consists in determining the pose of the platform for a given set of actuated joint coordinates (a given vector \mathbf{q}). This solution is needed for control purposes, calibration, and motion planning.

The forward kinematic problem of a parallel robot is usually much more complex than the inverse kinematic problem. Indeed, the loop closure equations (12.1) are typically highly nonlinear expressions of the pose variables. They form a nonlinear set of equations that generally admits multiple solutions (for example the Gough platform can have up to 40 solutions [12.11–13], while a table with the number of solutions for Gough platforms with special geometries is provided in [12.14]). The forward kinematic problem arises in two different contexts. In the first case, no estimate of the current pose of the platform is available (for example, when starting the robot) while in the second case, a relatively precise estimate of the pose is known (for example in real-time control when the forward kinematics has been solved at the previous sampling time of the controller). In the first case, the only known approach is to determine all the solutions of the inverse kinematic equations, although there is no known algorithm to sort the set of solutions. It is often possible to determine an upper bound on the number of real solutions. Consider for example the case of the planar 3- \underline{RPR} robot (Fig. 12.3). If the joint at point B_3 is disassembled, two separate mechanisms are obtained, namely a four-bar linkage and a rotary lever. From the kinematics of four-bar linkages it is known that point B_3 moves on a sixth-order algebraic curve. Moreover, it is also known that point B_3 of the lever moves on a cir-

cle, i.e., a second-order algebraic curve. For a given set of actuated joint coordinates, solutions to the forward kinematic problem arise when the two curves intersect, i.e., when the mechanism can be assembled. Bezout's theorem states that algebraic curves of order m, n intersect at nm points, counting their order of multiplicity. In the above case, this means that the curves intersect in 12 points. However, these 12 points include the two circular imaginary points that belong to the coupler curve of the four-bar linkage and to any circle, thereby to their intersection. These points are counted three times in Bezout's theorem and hence the forward kinematic problem will have at most six real solutions, corresponding to the intersection points.

Various methods have been proposed for the solution of the forward kinematic problem: elimination [12.15], continuation [12.13], Gröbner bases [12.16], and interval analysis [12.17]. Elimination is usually not very stable numerically (i.e., it can produce spurious roots and miss solutions) unless special care is taken in the implementation of the resulting univariate equation and the elimination steps. For instance, the transformation of polynomial solutions into eigenvalue problems can be used [12.18]. Polynomial continuation, on the other hand, is much more stable numerically since mature algorithms can be found in the literature [12.19]. The fastest methods – although not appropriate for real-time use – are Gröbner bases and interval analysis. They also have the advantage of being numerically certified (no roots can be missed and the solution can be computed with an arbitrarily prescribed accuracy).

However, in the simplest cases, elimination can usually be used to produce stable implementations. Consider, for example, the case of the planar 3- \underline{RPR} robot (Fig. 12.3), where the fixed reference frame has its origin at point A_1 and its x -axis passes through the point A_2 . Similarly, point B_1 is chosen as the origin of the moving frame while the x -axis of this frame passes through point B_2 . The pose of the platform is then defined by the coordinates (x, y) of point B_1 in the fixed reference frame and by the angle θ between the x -axes of the reference and moving frames. The known length q_1 of link 1 can thus be written simply as

$$q_1^2 = x^2 + y^2. \quad (12.2)$$

Then, the length q_i of the other links can be written as:

$$q_i^2 = x^2 + y^2 + g_i(x, y, \theta), \quad i = 2, 3, \quad (12.3)$$

where g_i is linear in x, y . Subtracting the length of link 1 from the latter two equations leads to a linear system of

two equations in the unknowns x and y . This system is solved and x and y are obtained as functions of θ . The expressions for x and y are then substituted into (12.2), thereby leading to a single equation in the unknown θ . This unknown appears in the equation through its sine and cosine, and by using the Weierstrass substitution, the equation can be transformed into a sixth-order polynomial equation in the unknown $T = \tan(\theta/2)$. Solving this equation in T allows one to compute all possible solutions in θ , from which the values of x and y are readily obtained. It is noted that this polynomial is of minimal degree since it was established that the number of real solutions cannot exceed six. It can also be shown that there exist configurations of the robot for which there are exactly six real solutions to the forward kinematics. Finally, it should be pointed out that the forward kinematic problem of the 3- \underline{RRR} robot is mathematically exactly the same as the one developed above for the 3- \underline{RPR} robot. This is typical for parallel robots.

When a priori information (an initial guess of the solution) is available, the forward kinematics is usually solved using the Newton–Raphson or the Newton–Gauss iterative scheme. Consider the solution to the inverse kinematic problem, written as:

$$\mathbf{q} = \mathbf{f}(\mathbf{x}) . \quad (12.4)$$

Iteration k of the Newton–Raphson procedure is written as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{A}(\mathbf{q} - \mathbf{f}(\mathbf{x}_k)) , \quad (12.5)$$

where \mathbf{q} is the vector of prescribed joint variables. The matrix \mathbf{A} is usually chosen as $(\frac{\partial \mathbf{f}}{\partial \mathbf{q}})^{-1}(\mathbf{x}_k)$ (the inverse matrix may not be computed at each iteration or even be chosen as constant). The iterative scheme stops when the magnitude of the difference vector $(\mathbf{q} - \mathbf{f}(\mathbf{x}_k))$ is smaller than a chosen threshold.

The choice of the inverse kinematics equations plays an important role in the convergence of this method [12.20]. For example, for a Gough platform, a minimal set of equations (six equations involving six

unknowns: three for the translation and three orientation angles) can be used, but other representations are possible. For instance, the position coordinates of three of the platform anchor points – in the fixed reference frame – can be used as unknowns (the coordinates of the remaining three points are then readily computed upon convergence). Using this representation, nine equations are needed. Six equations are obtained based on the known distances between the anchor points on the base and platform and three additional equations are obtained based on the known distances between all combinations of the three selected platform anchor points.

Provided that a good initial estimate of the solution is available, the Newton–Raphson algorithm is usually very fast. However, the procedure may not converge or, even worse, it may converge to a solution that is not the correct pose of the platform, i. e., it may converge to another assembly mode. Such a situation may occur even if the initial guess is arbitrarily close to the correct pose. If the result is used in a control loop, the consequences can be catastrophic. Fortunately mathematical tools such as the Kantorovitch theorem combined with interval analysis can be used to determine if the solution found by the Newton–Raphson scheme is the correct pose of the robot, i. e., to certify the result at the cost of a larger computation time, yet still compatible with real time [12.17].

Another possible approach for the solution of the forward kinematic problem is either to add sensors on the passive joints (e.g., on the U joints of a Gough platform) or to add passive legs with sensed joints. The main issue is then to determine the number and location of the additional sensors that will lead to a unique solution [12.21, 22] and to determine the effect of the sensor errors on the positioning error of the platform. For example, Stoughton mentions that, for a Gough platform with sensors in the U joints, it was still necessary to use the Newton–Raphson scheme to improve the accuracy of the solution because the solutions obtained with the additional sensors were very sensitive to measurement noise [12.23].

12.4 Velocity and Accuracy Analysis

Similarly to serial robots, the actuated joint velocity vector of parallel robots $\dot{\mathbf{q}}$ is related linearly to the vector of translational and angular velocities of the platform (for simplicity, the latter vector is denoted here by $\dot{\mathbf{p}}$, although the angular velocity is not the time derivative

of any set of angular parameters). The linear mapping between the two vectors is provided by the Jacobian matrix \mathbf{J} :

$$\dot{\mathbf{p}} = \mathbf{J}(\mathbf{p})\dot{\mathbf{q}} . \quad (12.6)$$

However, for parallel robots, a closed-form expression is usually available for the inverse Jacobian \mathbf{J}^{-1} , but this is much more difficult for \mathbf{J} (more precisely, the closed form for most six-DOF robots will be so complex that it cannot be used in practice). For example, a simple static analysis can be used to show that the i th row \mathbf{J}_i^{-1} of matrix \mathbf{J}^{-1} for a Gough platform can be written as

$$\mathbf{J}_i^{-1} = \mathbf{n}_i^\top (\mathbf{c}_i \times \mathbf{n}_i)^\top, \quad (12.7)$$

where \mathbf{n}_i represents the unit vector defined along leg i and \mathbf{c}_i is the vector connecting the origin of the mobile frame attached to the platform to the i th anchor point on the platform.

The effect of the joint sensor errors $\Delta \mathbf{q}$ on the positioning error $\Delta \mathbf{p}$ follows the same relationship, namely

$$\Delta \mathbf{p} = \mathbf{J}(\mathbf{p}) \Delta \mathbf{q}. \quad (12.8)$$

Since the matrix \mathbf{J} is difficult to obtain in closed form, accuracy analysis (i.e., finding the maximal positioning errors over a given workspace for bounded joint sensor errors) is much more difficult than for serial robots [12.20, 24]. Apart from the measurement errors, there are other sources of inaccuracy in parallel robots, namely: clearance in the passive joints, manufacturing tolerances, thermal errors, and gravity-induced and dynamic errors [12.25, 26]. The effect of joint clearances on trajectories followed by serial and parallel robots was studied in [12.27–29]. According to these works, it

is impossible to determine trends for the effect of the geometric errors: a case-by-case study must be performed since the effect is highly dependent on the mechanical architecture, dimensioning, and workspace of the robot. Thermal effects are sometimes mentioned as possible sources of inaccuracy, although few works substantiate this claim [12.30] and cooling may slightly reduce their effects [12.31].

Calibration is another means of improving the accuracy of parallel robots. This issue was addressed in Chap. 3. The methods and procedures used for parallel robots differ slightly from the ones used for serial robots since only the inverse kinematic equations are available and since the positioning of the platform is much less sensitive to the geometric errors than in serial robots [12.32, 33]. Hence the measurement noise occurring during calibration has a significant impact and may lead to surprising results. For example, the classical least-squares method may lead to parameters that are such that some constraint equations are not satisfied even when the measurement noise is taken into account [12.34]. It has also been shown with experimental data that classical parallel robot modeling leads to constraint equations that do not have any solution irrespective of the measurement noise [12.35]. Moreover, calibration is very sensitive to the choice of the calibration poses [12.36]: it seems that the optimal choice is to select poses on the workspace boundary [12.37, 38].

12.5 Singularity Analysis

12.5.1 General Formulation

The analysis of singularities of parallel mechanisms was first addressed by *Gosselin and Angeles* [12.39]. In this formulation, the kinematic equations were reduced to the input–output relationship between the actuated joint coordinate vector \mathbf{q} and the platform Cartesian coordinate vector \mathbf{p} , namely

$$f(\mathbf{q}, \mathbf{p}) = 0. \quad (12.9)$$

Differentiating (12.9) with respect to time leads to

$$\mathbf{B}\dot{\mathbf{q}} + \mathbf{A}\dot{\mathbf{p}} = \mathbf{0}. \quad (12.10)$$

Three types of singularities can then be defined:

- when matrix \mathbf{B} is singular (termed serial singularity)
- when matrix \mathbf{A} is singular (termed parallel singularity)

- when the input–output equations degenerate (termed an architecture singularity) in which case matrices \mathbf{A} and \mathbf{B} can both be singular.

In a serial singular configuration, the joints can have a nonzero velocity while the platform is at rest. In a parallel singularity, there exist nonzero platform velocities for which the joint velocities are zero. In the neighborhood of such a configuration, the robot can undergo an infinitesimal motion while the actuators are locked. Since the mobility of the end-effector should be zero when the actuators are locked, it is also said that in such a configuration the robot *gains* some DOFs. As a consequence, certain degrees of freedom of the platform cannot be controlled, and this is a major problem.

A more general study of singularities was proposed by *Zlatanov* [12.40]. The latter analysis uses the velocity equations involving the full twist of the end-effector and

all joint velocities (passive or actuated). This approach led to a more detailed classification of singularities and was also used later to reveal special singularities (referred to as *constraint singularities*) [12.41] that could not be found with the analysis presented in [12.39].

The singularity analyses discussed above are of the first order. Second- (and higher-) order singularity analyses can also be performed, although the latter are much more complex [12.29, 42].

12.5.2 Parallel Singularity Analysis

This type of singularity is especially important for parallel robots because it corresponds to configurations in which the control of the robot is lost. Furthermore, very large joint forces can occur in the vicinity of singular poses, that may lead to a breakdown of the robot. The main issues to be addressed in this context are:

- the characterization of the singularities
- the definition of performance indices representing the closeness to a singularity
- the development of algorithms that are capable of detecting the presence of singularities over a given workspace or trajectory

Parallel singularities arise when the 6×6 full inverse Jacobian \mathbf{J}_f (i.e., the matrix that maps the full twist of the platform into active – and eventually passive – joint velocities) is singular, i.e., when its determinant $\det(\mathbf{J}_f)$ is equal to 0. It is pointed out that passive joint velocities sometimes have to be included because restricting the analysis to the active joint velocities may not allow the determination of all singular configurations of the mechanism (see the example of the 3 – UPU robot [12.43]). Usually, a proper velocity analysis allows one to establish this matrix in closed form, but computing the determinant of this matrix may be difficult even with symbolic computation tools (see the example of the Gough platform [12.44, 45]). The advantage of this approach is that once the expression of the determinant is obtained, the locus of singular configurations can be plotted in the workspace, thereby leading to graphical representations that can be useful in a context of design. However, the determinant itself is usually a large expression that does not provide insight into the geometric conditions associated with the singularities.

An alternative approach is to use line geometry: indeed, for several parallel robots (although not all), a row of \mathbf{J}_f corresponds to the Plücker vector of some line defined on the links of the robot. For example, for a Gough platform, the rows of \mathbf{J}_f are the normalized Plücker

vectors of the lines associated with the legs of the robot. A singularity of matrix \mathbf{J}_f therefore implies a linear dependency between these vectors (they then constitute a *linear complex*), a situation that may occur only if the lines associated with the vectors verify particular geometric constraints [12.46] (for example, three Plücker vectors will be linearly dependent if and only if their associated lines are coplanar and intersect at a common point). These geometric constraints have been identified by Grassmann for every set of three, four, five or six vectors. Singularity analysis is thus reduced to determining conditions on the pose parameters for which these constraints are satisfied, giving geometric information on the singularity variety. The closed-form singularity condition can then be substituted into the Jacobian matrix and the kernel of the matrix calculated in order to determine the singular motion [12.47].

Measuring closeness between a pose and a singular configuration is a difficult problem: there exists no mathematical metric defining the distance between a prescribed pose and a given singular pose. Hence, a certain level of arbitrariness must be accepted in the definition of the *distance* to a singularity and none of the proposed indices is perfect. For example, a possible index is the determinant of \mathbf{J}_f : unfortunately when the platform is subjected to both translational and rotational motion, the latter matrix is not dimensionally homogeneous and hence the value of the determinant will change according to the physical units used to describe the geometry of the robot. Dexterity indices, as defined in Chap. 11 (although they are less relevant for parallel robots than for serial robots [12.24]), can also be used and special indices for parallel robots have also been proposed [12.48, 49].

Most of the analyses based on the above indices are local, i.e., valid only for a given pose, while in practice the problem is to determine if a singular configuration can occur over a given workspace or trajectory. Fortunately, an algorithm exists that allows this verification, even if the geometric model of the robot is uncertain [12.50]. However, it should be pointed out that a singularity-free workspace may not always be optimal for parallel robots. Indeed, other performance requirements may impose the presence of singularities in the workspace or, the robot may have part of its workspace singularity-free (for example its working region) while exhibiting singularities outside this region. Therefore, a motion planner producing a trajectory that avoids singularities while remaining close to a desired path is advisable and various approaches have been proposed to address this problem [12.51, 52].

Finally, it is noted that being close to a singular configuration may be useful in some cases. For example, large amplification factors between the end-effector motion and the actuated joint motion may be interesting for fine-positioning devices with a very small workspace or for improving the sensitivity along some

measurement directions for a parallel robot used as a force sensor [12.53]. It should also be mentioned that parallel robots that remain *permanently* in a singular configuration may be interesting since they are capable of producing complex motions with only one actuator [12.54–56].

12.6 Workspace Analysis

As mentioned in the introduction, one of the main drawbacks of parallel robots is their small workspace. Moreover, their workspace analysis is usually more complex than for serial robots, especially if the platform has more than 3 DOFs. Indeed, the platform motion capabilities are usually coupled, thereby prohibiting simple graphical representations of the workspace. *Decoupled robots* have also been proposed [12.57–59] but they do not provide the load-carrying capacity of conventional parallel robots. In general, the workspace of parallel robots is restricted by

- the limitations on the actuated joint variables: for example, the length of the legs of a Gough platform have a minimum and maximum value,
- the limitations on the range of motion of the passive joints: for example the U and S joints of a Gough platform have restricted ranges of angular motion,
- the mechanical interferences between the bodies of the mechanism (legs, base, and platform).

Different types of workspaces can be defined such as the *dextrous* workspace (the set of platform locations in which all rotations are possible), the *maximal* workspace (the set of platform locations that may be reached with at least one orientation) or the *orientation* workspace (the set of orientations that can be reached for a given location of the platform).

A possible approach for representing the workspace of parallel robots with $n > 3$ DOFs is to fix the value of $n - 3$ pose parameters in order to be able to plot the workspace over the remaining three DOFs. Such plots can be obtained very efficiently using geometric methods if the plotted variables involve only translational motions because a geometric approach usually allows one to establish the nature of the boundary of the workspace (see, for example, [12.60, 61] for the calculation of the workspace of a Gough platform when its orientation is fixed). Another advantage of this approach is that it allows the computation of the surface and volume of the workspace while being very efficient

in terms of storage space. However, when rotational motion is included, the geometric approach becomes rather complex. Possible alternatives include:

- discretization methods in which all poses of a n -dimensional grid are checked with respect to the kinematic constraints. Such methods are computationally intensive because the computation time increases exponentially with the resolution of the mesh. Also they require a large storage space. On the other hand, the advantage of discretization is that it is usually simple to implement and it allows one to take into account all kinematic constraints as they can usually be simply verified for a given pose,
- numerical methods that allow the determination of the workspace boundary [12.62, 63],
- numerical methods based on interval analysis that allow the determination of an approximation of the workspace volume up to an arbitrary accuracy [12.64]. This representation is also appropriate for motion planning problems.

Singularities can also split the workspace calculated from the above kinematic constraints into elementary components, called *aspects* by Wenger [12.65] and which are separated by a singularity variety. However, determining the aspects for spatial robots is still an open problem. A parallel robot may not always be able to move from one aspect to another (at least without considering the dynamics of the robot [12.66]) and hence the useful workspace can be reduced.

A problem related to workspace analysis is the *motion planning problem*, which is slightly different from that encountered with serial robots. For parallel robots, the problem is not to avoid obstacles within the workspace but to determine if a given trajectory lies entirely inside the workspace of the robot or to determine a trajectory between two poses that lies in the workspace and is singularity-free. Algorithms for checking if a trajectory is valid are available [12.67] but finding such a trajectory is more difficult. Indeed, classical serial

robot motion planners work in the joint space and assume that there is a one-to-one relationship between the joint space and the operational space. With this assumption, it is possible to determine a set of points in the joint space that is collision-free and to use this knowledge to build a collision-free path between two poses. However, this assumption is not valid for parallel robots because the mapping between the joint space and the operational space is not one-to-one: a point in the joint space may either correspond to multiple points in the operational space or have no correspondence because the closure equations of the mechanism are not satisfied. For parallel robots, the most efficient motion planner seems to be an adaptation of a probabilistic motion planner that takes into account – to a certain extent – the closure equations [12.68, 69].

Another motion planning problem is related to tasks where a limited number of DOFs of the robot are used. For example, when a six-DOF parallel robot is used for machining, the rotation of the platform around the axis of the tool is not used since the spindle rotation ensures the needed motion. Thus, an additional DOF is available that can be used to increase the machine's workspace, to avoid singularities or to optimize some performance index of the robot [12.70, 71]. Along the same lines, it is possible to define the *part positioning* problem [12.72] as determining the pose of a rigid body with respect to the workspace of the robot so that this pose satisfies some constraints (e.g., the rigid body should lie fully inside the workspace and at all points of its surface the robot should have some rotational ability).

12.7 Static Analysis and Static Balancing

Similarly to serial robots, the static analysis of parallel robots can be readily performed using the Jacobian matrix. Indeed, the mapping between the vector of actuated joint forces/torques $\boldsymbol{\tau}$ and the external wrench \boldsymbol{f} exerted by the platform can be written as

$$\boldsymbol{\tau} = \boldsymbol{J}^\top \boldsymbol{f}, \quad (12.11)$$

where \boldsymbol{J}^\top is the transpose of the Jacobian matrix of the robot. Equation (12.11) can be used for various purposes, namely

- during the *design* process in order to determine the actuator forces/torques (for actuator selection). In this case, the designer is interested in finding the maximal actuator forces/torques over the robot workspace. This is a complex issue since \boldsymbol{J}^\top is not known in closed form,
- in applications where the robot is used as a *force sensor*: if $\boldsymbol{\tau}$ is measured and the pose of the platform is known, then (12.11) can be used to calculate \boldsymbol{f} so that the robot can be used both as a motion and sensing platform [12.73–75].

The stiffness matrix \boldsymbol{K} of a parallel robot is defined similarly to that of serial robots as

$$\boldsymbol{K} = \boldsymbol{J}^{-\top} \boldsymbol{K}_j \boldsymbol{J}^{-1}, \quad (12.12)$$

where \boldsymbol{K}_j is the diagonal matrix of actuated joint stiffness. Duffy [12.76] notes that this derivation is, in general, incomplete. For example, for a Gough platform

this derivation assumes that there is no initial load on the elastic element of the link. Assuming that the length of the unloaded link is q_i^0 leads to

$$\begin{aligned} \Delta \boldsymbol{f} &= \sum_{i=1}^{i=6} k \Delta q_i \boldsymbol{n}_i + k_i (q_i - q_i^0) \Delta \boldsymbol{n}_i, \\ \Delta \boldsymbol{m} &= \sum_{i=1}^{i=6} k \Delta q_i \boldsymbol{c}_i \times \boldsymbol{n}_i + k_i (q_i - q_i^0) \Delta (\boldsymbol{c}_i \times \boldsymbol{n}_i), \end{aligned}$$

where k_i denotes the axial stiffness of the leg, \boldsymbol{n}_i represents the unit vector defined along the i th leg, \boldsymbol{c}_i is the vector connecting the origin of the moving reference frame to the i th anchor point of the platform, and \boldsymbol{f} and \boldsymbol{m} are the external force and moment vectors that are applied by the platform. Consequently, the stiffness matrix as defined in (12.12) is valid only if $q_i = q_i^0$, and is coined the *passive stiffness*.

Moreover, it was pointed out in the literature [12.77] that the formulation of (12.12) is valid only when the external wrench is zero. Indeed, the above formulation does not account for the fact that the Jacobian matrix is configuration dependent and therefore changes with the applied loads. The formulation proposed in [12.77] and termed the *conservative congruence transformation* (CCT) takes these changes into account and leads to a stiffness matrix defined as

$$\boldsymbol{K}_c = \boldsymbol{J}^{-\top} \boldsymbol{K}_j \boldsymbol{J}^{-1} + \left(\frac{\partial \boldsymbol{J}^{-\top} \boldsymbol{\tau}}{\partial \boldsymbol{p}} \right), \quad (12.13)$$

where \mathbf{p} is the vector of Cartesian coordinates and $\boldsymbol{\tau}$ is the vector of actuated joint forces/torques. Whenever possible, the formulation of (12.13) should be used instead of that of (12.12) because it is mechanically consistent.

Another interesting static problem is the static balancing of parallel robots. The static balancing of mechanisms in general has been an important research topic for several decades (see, for instance, [12.78] for an account of the state of the art and many recent new results). Parallel mechanisms are said to be *statically balanced* when the weight of the links does not produce any torque (or force) at the actuators under static conditions, for any configuration of the manipulator or mechanism. This condition is also referred to as *gravity compensation*. The gravity compensation of parallel mechanisms was addressed by Dunlop [12.79] who suggested the use of counterweights to balance a two-DOF parallel robot used for antenna aiming, and by Jean [12.80] for planar robots. The latter case leads to simple and elegant balancing conditions.

In general, static balancing can be achieved using counterweights and/or springs. When springs are used, static balancing can be defined as the set of conditions for which the total potential energy in the mechanism – including gravitational energy and the elastic energy stored in the springs – is constant for any configuration of the mechanism. When no springs – or other means of storing elastic energy – are used, then static balancing conditions imply that the center of mass of the mechanism does not move in the direction of the gravity vector, for any motion of the mechanism.

Consider a general spatial n -degree-of-freedom parallel mechanism composed of n_b moving bodies and one fixed link. Moreover, let the position vector of the center of mass of each moving body with respect to a fixed reference frame be noted \mathbf{c}_i and let the mass of the i th moving body be noted m_i . The position vector of the center of mass of the mechanism with respect to the fixed frame, noted \mathbf{c} , can be written as

$$\mathbf{c} = \frac{1}{M} \sum_{i=1}^{n_b} m_i \mathbf{c}_i, \quad (12.14)$$

where M is the total mass of the moving links, i. e.,

$$M = \sum_{i=1}^{n_b} m_i. \quad (12.15)$$

In general, the vector \mathbf{c} is a function of the configuration of the mechanism, i. e.,

$$\mathbf{c} = \mathbf{c}(\boldsymbol{\theta}), \quad (12.16)$$

where $\boldsymbol{\theta}$ is the vector comprising all the joint coordinates of the mechanism.

Following this notation, and if no elastic components are used, the condition for static balancing can be written as

$$\mathbf{e}_z^\top \mathbf{c} = C_t, \quad (12.17)$$

where C_t is an arbitrary constant and \mathbf{e}_z is a unit vector oriented in the direction of gravity.

When elastic components are used, the total potential energy in the mechanism, denoted by V , is defined as the sum of the gravitational and elastic potential energy and can be written as

$$V = g \mathbf{e}_z^\top \sum_{i=1}^{n_b} m_i \mathbf{c}_i + \frac{1}{2} \sum_{j=1}^{n_s} k_j (s_j - s_j^0)^2, \quad (12.18)$$

where g is the magnitude of the gravitational acceleration, n_s is the number of linear elastic components in the system, k_j is the stiffness of the j th elastic component, s_j is the length of the j -th elastic component, and s_j^0 is its undeformed length. As mentioned above, when elastic elements are used, the condition for static balancing is that the total potential energy is constant, which can be written as

$$V = V_c, \quad (12.19)$$

where V_c is an arbitrary constant.

The general equations given above can be used to obtain the conditions under which a given mechanism is statically balanced. In general, the equations provide sufficient conditions for balancing. The problem of statically balancing spatial parallel robots is usually complex. Several papers were published recently on this subject [12.81, 82]. Among other results, it was pointed out that it is not possible to balance a 6-UPS robot in all poses by using only counterweights [12.83]. Alternative mechanisms (using parallelograms) that can be balanced using only springs were suggested [12.81, 84, 85].

A natural extension of static balancing is dynamic balancing. The dynamic balancing problem consists in finding parallel mechanisms that do not produce any reaction force or torque on their base for arbitrary trajectories. This problem was addressed for planar and spatial parallel mechanisms. In [12.86], it was shown that dynamically balanced parallel mechanisms with up to six DOFs can be synthesized, although the resulting mechanical architectures may be complex.

12.8 Dynamic Analysis

The dynamic model of parallel robots has a form similar to that of serial robots (see Chap. 2), namely

$$\mathbf{M}(\mathbf{x})\ddot{\mathbf{p}} + \mathbf{C}(\dot{\mathbf{p}}, \dot{\mathbf{q}}, \mathbf{p}, \mathbf{q}) + \mathbf{G}(\mathbf{p}, \mathbf{q}) = \boldsymbol{\tau}, \quad (12.20)$$

where \mathbf{M} is the positive-definite generalized inertia matrix, \mathbf{G} is the gravitational term, and \mathbf{C} is the centrifugal and Coriolis term. However, this equation is more difficult to obtain for parallel robots because the closure equations have to be satisfied. A classical method for calculating the dynamic models of closed-chain mechanisms is to consider first an equivalent tree-structure, and then to enforce the kinematic constraints using Lagrange multipliers or d'Alembert's principle [12.87,88], a principle that has proved to be valid in [12.89]. Other approaches include the use of the principle of virtual work [12.90–93], the Lagrange formalism [12.94–96], Hamilton's principle [12.97], and Newton–Euler equations [12.98–103].

The main drawbacks of dynamic models obtained for parallel robots is that they are usually rather complex, they require the determination of dynamic parameters that are often not well known, and they involve solving the forward kinematics. Therefore, their calculation

is computer intensive while they must be used in real time.

Using dynamic models for control purposes was proposed [12.104, 105], usually in the context of an adaptive control scheme, in which the tracking errors are used online to correct the parameters used in the dynamic equations [12.106, 107]. Control laws were proposed mainly for planar robots and for the Delta robot [12.108–110] although some implementations have been proposed for general six-DOF robots [12.111, 112] or vibration platforms [12.113]. However for six-DOF robots, the benefits of using dynamic models for high-speed motions are difficult to establish because the computational burden of the dynamic model somewhat reduces the potential gains.

This is clearly an important and open issue since parallel robots can operate at velocities and accelerations much larger than those of serial robots. For example, some Delta robots have reached accelerations of the order of 500 m/s² while wire-driven robots can probably exceed this value.

Finally, the optimization of the dynamic properties of parallel mechanisms can be addressed by minimizing the variation of the inertia over the workspace [12.114].

12.9 Design

The design of a robot can be decomposed into two main phases:

- *structural synthesis*: finding the general mechanical structure of the robot
- *geometric synthesis*: determining the value of the geometric parameters involved in a given structure (here geometric parameters must be understood in a loose sense, for example, mass and inertia may also be involved)

The problem of structural synthesis (type synthesis) was addressed in Sect. 12.2. However, performance requirements other than the motion pattern of the robot have to be taken into account in the design of a robot. Serial robots have the advantage that the number of possible mechanical architectures is relatively small and that some of these architectures have clear advantages in terms of performance compared to others (for example, the workspace of a 3R structure is much larger than the workspace of a Cartesian robot of similar size).

Unfortunately no such rules exist for parallel robots, for which there are furthermore a large number of possible mechanical designs. Additionally, the performances of parallel robots are very sensitive to their geometric parameters. For example, the extremal stiffness of a Gough platform over a given workspace can change by 700% for a change of only 10% of the platform radius. Consequently, structural synthesis cannot be dissociated from the geometric synthesis. In fact, it is conjectured that a well-dimensioned robot of any structural type will in general perform better than a poorly designed robot with a structure that may seem more appropriate for the task at hand.

Usually the design process is treated as an optimization problem. To each specified performance requirement is associated a performance index whose value increases with the level of violation of the performance requirement. These performance indices are summed in a weighted real-valued function called the *cost function*, which is a function of the geometric design parameters

and then a numerical optimization procedure is used to find the parameters that minimize the cost function (hence this approach leads to what is called an *optimal design*) [12.115–117]. There are however numerous drawbacks to this approach: the determination and effect of the weights in the cost function are difficult to ascertain, imperative requirements are difficult to incorporate and make the optimization process more complex, and the definition of the performance indices is not trivial, to name but a few. The main issues can be stated as:

- the robustness of the design solution obtained with the cost function approach with respect to the uncertainties in the final design. Indeed the real instantiation of a theoretical solution will always differ from the latter because of the manufacturing tolerances and other uncertainties that are inherent to a mechanical system,
- performance requirements may be antagonistic (e.g., workspace and accuracy) and the optimal design approach only provides a compromise between these requirements that is difficult to master through the weights.

An alternative to optimal design is referred to as *appropriate design*, in which no optimization is considered but the objective is to ensure that desired requirements are satisfied. This approach is based on the definition of the *parameter space* in which each dimension is associated with a design parameter. Performance requirements are considered in turn and the regions of the parameter space that correspond to robots satisfying the requirements are calculated. The design solution is then obtained as the intersection of the regions obtained for each individual requirement.

In practice, only an approximation of the regions is necessary since values close to the boundary cannot be physically realized due to manufacturing tolerances. For that calculation, interval analysis was successfully used in various applications [12.20, 118].

The appropriate design approach is clearly more complex to implement than the cost function approach but has the advantage of providing all design solutions, with the consequence that manufacturing tolerances may be taken into account to ensure that the real robot will also satisfy the desired requirements.

12.10 Application Examples

Parallel robots have been successfully used in many applications, a few of which will now be briefly mentioned. Almost all recent land-based telescopes use parallel robots, either as a secondary mirror alignment system (for example the University of Arizona MMT or the European Organisation for Astronomical Research in the Southern Hemisphere (ESO) Visible and Infrared Survey Telescope for Astronomy (VISTA)) or as a primary mirror pointing device. A wire-driven parallel robot flew in the space shuttle mission STS-63 in February 1999 while an octopod (a parallel robot with eight legs) was used to isolate the space shuttle payload from vibration. A parallel platform is used in the medical rehabilitation platform Caren of Motek. All flight simulators use a parallel structure as a motion platform. They are nowadays used for driving simulators as well [12.119].

In industry, numerous machine tools based on parallel structures have been designed: some of them have found a niche market (e.g., the Tricept) and it can be expected that more will be used in the future (especially for high-speed manufacturing) as soon as a controller specifically designed for parallel structures is developed (current controllers are basically identical to the ones used for the classical linear machine tools and hence do not allow the potential of parallel structures to be fully exploited). Ultra-accurate positioning devices based on parallel robots are proposed by companies such as Physik Instrumente, Micos, and Aljo. In the food industry the Delta robot proposed by Demarex and SIG Robotics is widely used for fast packaging. Other companies such as Adept are also developing fast parallel robots with three and four degrees of freedom.

12.11 Conclusion and Further Reading

The analysis of a parallel mechanism may be partly based on methods that are presented in other chapters of this handbook:

- *kinematics*: kinematics background is covered in Chap. 1,
- *dynamics*: general approaches are presented in Chap. 2 while identification of the dynamic parameters is addressed in Chap. 14,
- *design*: design methodologies are covered in Chap. 10,
- *control*: control issues are presented in Chaps. 5, 6, and 7 although the closed-loop structure of parallel robots may require some adaptation of the control schemes.

It must also be emphasized that efficient numerical analysis is a key point for many algorithms related to parallel robots. System solving with Gröbner basis, continuation method, and interval analysis is essential for kinematics, workspace, and singularity analysis.

Further information and up-to-date extensive references and papers related to parallel robots can be found on the following two web sites [12.120, 121]

Useful complementary readings on parallel robots are [12.9, 122, 123].

Parallel robots are slowly finding their way into various applications, not only in industry but also in field and service robotics, as discussed in Part F of this handbook. Still, compared to their serial counterpart, their analysis is far from being complete.

References

- 12.1 J.E. Gwinnett: Amusement device, Patent 1789680 (1931)
- 12.2 V.E. Gough: Contribution to discussion of papers on research in automobile stability, control and tyre performance, 1956–1957. *Proc. Auto Div. Inst. Mech. Eng.*
- 12.3 D. Stewart: A platform with 6 degrees of freedom, *Proc. Inst. Mech. Eng.* **180**(1,15), 371–386 (1965)
- 12.4 I.A. Bonev: The true origins of parallel robots. <http://www.parallemic.org/Reviews/Review007.html>, (2003)
- 12.5 J. Albus, R. Bostelman, N. Dagalakis: The NIST ROBOCRANE, *J. Robot. Syst.* **10**(5), 709–724 (1993)
- 12.6 R. Clavel: DELTA, a fast robot with parallel geometry. In: *18th Int. Symp. on Industrial Robots (ISIR)*, Lausanne, (1988) pp. 91–100
- 12.7 K.H. Hunt: Structural kinematics of in parallel actuated robot arms, *J. Mech. Transmiss. Automation Design* **105**(4), 705–712 (1983)
- 12.8 J.M. Hervé: Group mathematics and parallel link mechanisms. In: *9th IFTOMM World Congress on the Theory of Machines and Mechanisms*, Milan, (1995) pp. 2079–2082
- 12.9 X. Kong, C.M. Gosselin: *Type synthesis of parallel mechanisms* (Springer Tracts in Advanced Robotics, Heidelberg 2007), Vol. 33
- 12.10 www-sop.inria.fr/coprin/equipe/merlet/Archi/archirobot.html
- 12.11 P. Dietmaier: The Stewart–Gough platform of general geometry can have 40 real postures. In: *ARK*, Strobl, (1998) pp. 7–16
- 12.12 M.L. Husty: An algorithm for solving the direct kinematic of Stewart–Gough-type platforms, *Mechanism Machine Theory* **31**(4), 365–380 (1996)
- 12.13 M. Raghavan: The Stewart platform of general geometry has 40 configurations, *ASME J. Mech. Des.* **115**(2), 277–282 (1993)
- 12.14 J.C. Faugère, D. Lazard: The combinatorial classes of parallel manipulators, *Mechanism Machine Theory* **30**(6), 765–776 (1995)
- 12.15 T.-Y. Lee, J.-K. Shim: Improved dyalytic elimination algorithm for the forward kinematics of the general Stewart–Gough platform, *Mechanism Machine Theory* **38**(6), 563–577 (2003)
- 12.16 F. Rouillier: Real roots counting for some robotics problems. In: *Computational Kinematics*, ed. by J.-P. Merlet, B. Ravani (Kluwer, Dordrecht 1995) pp. 73–82
- 12.17 J.-P. Merlet: Solving the forward kinematics of a Gough-type parallel manipulator with interval analysis, *Int. J. Robot. Res.* **23**(3), 221–236 (2004)
- 12.18 D. Manocha: Algebraic and Numeric Techniques for Modeling and Robotics. Ph.D. Thesis (University of California, Berkeley 1992)
- 12.19 A.J. Sommese, C.W. Wampler: *Numerical Solutions of Polynomial Systems Arising in Engineering and Science* (World Scientific, Singapore 2005)
- 12.20 J.-P. Merlet, D. Daney: Dimensional synthesis of parallel robots with a guaranteed given accuracy over a specific workspace. In: *IEEE Int. Conf. on Robotics and Automation*, Barcelona, (2005)
- 12.21 L. Baron, J. Angeles: The direct kinematics of parallel manipulators under joint-sensor redundancy, *IEEE Trans. Robot. Automat.* **16**(1), 12–19 (2000)
- 12.22 J.-P. Merlet: Closed-form resolution of the direct kinematics of parallel manipulators using extra sensors data. In: *IEEE Int. Conf. on Robotics and Automation*, Atlanta, (1993) pp. 200–204

- 12.23 R. Stoughton, T. Arai: Kinematic optimization of a chopsticks-type micro-manipulator. In: *Japan-USA Symp. on Flexible Automation*, San Francisco, (1993) pp. 151–157
- 12.24 J.-P. Merlet: Jacobian, manipulability, condition number, and accuracy of parallel robots. *ASME J. Mech. Design* **128**(1), 199–206 (2006)
- 12.25 F.-T. Niarirsiry, N. Fazenda, R. Clavel: Study of the source of inaccuracy of a 3 dof flexure hinge-based parallel manipulator. In: *IEEE Int. Conf. on Robotics and Automation*, New Orleans, (2004) pp. 4091–4096
- 12.26 G. Pritschow, C. Eppler, T. Garber: Influence of the dynamic stiffness on the accuracy of PKM. In: *3rd Chemnitzer Parallelkinematik Seminar*, Chemnitz, (2002) pp. 313–333
- 12.27 V. Parenti-Castelli, S. Venanzi: On the joint clearance effects in serial and parallel manipulators. In: *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, Québec, (2002) pp. 215–223
- 12.28 A. Pott, M. Hiller: A new approach to error analysis in parallel kinematic structures. In: *ARK, Sestri-Levante*, (2004)
- 12.29 K. Wohlhart: Degrees of shakiness, *Mechanism Machine Theory* **34**(7), 1103–1126 (1999)
- 12.30 K. Tönshoff, B. Denkena, G. Günther, H.-C. Möhring: Modelling of error effects on the new hybrid kinematic DUMBO structure. In: *3rd Chemnitzer Parallelkinematik Seminar*, Chemnitz, (2002) pp. 639–653
- 12.31 U. Sellgren: Modeling of mechanical interfaces in a systems context. In: *Int. ANSYS Conf.*, Pittsburgh, (2002)
- 12.32 W. Khalil, S. Besnard: Identifiable parameters for the geometric calibration of parallel robots, *Arch. Control Sci.* **11**(3–4), 263–277 (2001)
- 12.33 C.W. Wampler, J.M. Hollerbach, T. Arai: An implicit loop method for kinematic calibration and its application to closed-chain mechanisms, *IEEE Trans. Robot. Autom.* **11**(5), 710–724 (1995)
- 12.34 D. Daney, Y. Papegay, A. Neumaier: Interval methods for certification of the kinematic calibration of parallel robots. In: *IEEE Int. Conf. on Robotics and Automation*, New Orleans, (2004) pp. 1913–1918
- 12.35 D. Daney, N. Andreff, Y. Papegay: Interval method for calibration of parallel robots: a vision-based experimentation. In: *Computational Kinematics*, Cassino, (2005)
- 12.36 A. Nahvi, J.M. Hollerbach: The noise amplification index for optimal pose selection in robot calibration. In: *IEEE Int. Conf. on Robotics and Automation*, Minneapolis, (1996) pp. 647–654
- 12.37 G. Meng, L. Tiemin, Y. Wensheng: Calibration method and experiment of Stewart platform using a laser tracker. In: *Int. Conf on Systems, Man and Cybernetics*, The Hague, (2003) pp. 2797–2802
- 12.38 D. Daney: Optimal measurement configurations for Gough platform calibration. In: *IEEE Int. Conf. on Robotics and Automation*, Washington, (2002) pp. 147–152
- 12.39 C. Gosselin, J. Angeles: Singularity analysis of closed-loop kinematic chains, *IEEE Trans. Robot. Automation* **6**(3), 281–290 (1990)
- 12.40 D. Zlatanov, R.G. Fenton, B. Benhabib: A unifying framework for classification and interpretation of mechanism singularities, *ASME J. Mech. Des.* **117**(4), 566–572 (1995)
- 12.41 D. Zlatanov, I.A. Bonev, C.M. Gosselin: Constraint singularities of parallel mechanisms. In: *IEEE Int. Conf. on Robotics and Automation*, Washington, (2002) pp. 496–502
- 12.42 G. Liu, Y. Lou, Z. Li: Singularities of parallel manipulators: a geometric treatment, *IEEE Trans. Robot. Autom.* **19**(4), 579–594 (2003)
- 12.43 I.A. Bonev, D. Zlatanov: The mystery of the singular SNU translational parallel robot. www.parallelic.org/Reviews/Review004.html, (2001)
- 12.44 H. Li., C.M. Gosselin, M.J. Richard, B. St-Onge Mayer: Analytic form of the six-dimensional singularity locus of the general Gough-Stewart platform, *ASME J. Mech. Des.* **128**(1), 279–287 (2006)
- 12.45 B. St-Onge Mayer, C.M. Gosselin: Singularity analysis and representation of the general Gough-Stewart platform, *Int. J. Robot. Res.* **19**(3), 271–288 (2000)
- 12.46 J.-P. Merlet: Singular configurations of parallel manipulators and Grassmann geometry, *Int. J. Robot. Res.* **8**(5), 45–56 (1989)
- 12.47 J.-P. Merlet: On the infinitesimal motion of a parallel manipulator in singular configurations. In: *IEEE Int. Conf. on Robotics and Automation*, Nice, (1992) pp. 320–325
- 12.48 H. Pottmann, M. Peternell, B. Ravani: Approximation in line space. Applications in robot kinematics. In: *ARK, Strobl*, (1998) pp. 403–412
- 12.49 P.A. Voglewede, I. Ebert-Uphoff: Measuring "closeness" to singularities for parallel manipulators. In: *IEEE Int. Conf. on Robotics and Automation*, New Orleans, (2004) pp. 4539–4544
- 12.50 J.-P. Merlet, D. Daney: A formal-numerical approach to determine the presence of singularity within the workspace of a parallel robot. In: *Computational Kinematics*, ed. by F.C. Park, C.C. Iurascu (EJCK, Seoul 2001) pp. 167–176
- 12.51 S. Bhattacharya, H. Hatwal, A. Ghosh: Comparison of an exact and an approximate method of singularity avoidance in platform type parallel manipulators, *Mechanism Machine Theory* **33**(7), 965–974 (1998)
- 12.52 D.N. Nenchev, M. Uchiyama: Singularity-consistent path planning and control of parallel robot motion through instantaneous-self-motion type. In: *IEEE*

- Int. Conf. on Robotics and Automation*, Minneapolis, (1996) pp. 1864–1870
- 12.53 R. Ranganath, P.S. Nair, T.S. Mruthyunjaya, A. Ghosal: A force-torque sensor based on a Stewart platform in a near-singular configuration, *Mechanism Machine Theory* **39**(9), 971–998 (2004)
- 12.54 M.L. Husty, A. Karger: Architecture singular parallel manipulators and their self-motions. In: *ARK*, Piran, (2000) pp. 355–364
- 12.55 A. Karger: Architecture singular planar parallel manipulators, *Mechanism Machine Theory* **38**(11), 1149–1164 (2003)
- 12.56 K. Wohlhart: Mobile 6-SPS parallel manipulators, *J. Robot. Syst.* **20**(8), 509–516 (2003)
- 12.57 C. Innocenti, V. Parenti-Castelli: Direct kinematics of the 6-4 fully parallel manipulator with position and orientation uncoupled. In: *European Robotics and Intelligent Systems Conf.*, Corfou, (1991)
- 12.58 G. Gogu: Mobility of mechanisms: a critical review, *Mechanism Machine Theory* **40**(10), 1068–1097 (2005)
- 12.59 I. Zabalza, J. Ros, J.J. Gil, J.M. Pintor, J.M. Jimenez: Tri-Scott. A new kinematic structure for a 6-dof decoupled parallel manipulator. In: *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, Québec, (2002) pp. 12–15
- 12.60 C. Gosselin: Determination of the workspace of 6-dof parallel manipulators, *ASME J. Mech. Des.* **112**(3), 331–336 (1990)
- 12.61 J.-P. Merlet: Geometrical determination of the workspace of a constrained parallel manipulator. In: *ARK*, Ferrare, (1992) pp. 326–329
- 12.62 F.A. Adkins, E.J. Haug: Operational envelope of a spatial Stewart platform, *ASME J. Mech. Des.* **119**(2), 330–332 (1997)
- 12.63 E.J. Haug, F.A. Adkins, C.M. Luh: Operational envelopes for working bodies of mechanisms and manipulators, *ASME J. Mech. Des.* **120**(1), 84–91 (1998)
- 12.64 J.-P. Merlet: Determination of 6D workspaces of Gough-type parallel manipulator and comparison between different geometries, *Int. J. Robot. Res.* **18**(9), 902–916 (1999)
- 12.65 P. Wenger, D. Chablat: Workspace and assembly modes in fully parallel manipulators: a descriptive study. In: *ARK*, Strobl, (1998) pp. 117–126
- 12.66 J. Hesselbach, C. Bier, A. Campos, H. Löwe: Parallel robot specific control functionalities. In: *2nd Int. Colloquium, Collaborative Research Centre 562*, Braunschweig, (2005) pp. 93–108
- 12.67 J.-P. Merlet: An efficient trajectory verifier for motion planning of parallel machine. In: *Parallel Kinematic Machines Int. Conf.*, Ann Arbor, (2000)
- 12.68 J. Cortés, T. Siméon: Probabilistic motion planning for parallel mechanisms. In: *IEEE Int. Conf. on Robotics and Automation*, Taipei, (2003) pp. 4354–4359
- 12.69 J.H. Yakey, S.M. LaValle, L.E. Kavraki: Randomized path planning for linkages with closed kinematic chains, *IEEE Trans. Robot. Autom.* **17**(6), 951–958 (2001)
- 12.70 J.-P. Merlet, M.-W. Perng, D. Daney: Optimal trajectory planning of a 5-axis machine tool based on a 6-axis parallel manipulator. In: *ARK*, Piran, (2000) pp. 315–322
- 12.71 D. Shaw, Chen Y.-S. Cutting path generation of the Stewart platform-based milling machine using an end-mill, *Int. J. Prod. Res.* **39**(7), 1367–1383 (2001)
- 12.72 Z. Wang, Z. Wang, W. Liu, Y. Lei: A study on workspace, boundary workspace analysis and workpiece positioning for parallel machine tools, *Mechanism Machine Theory* **36**(6), 605–622 (2001)
- 12.73 D.R. Kerr: Analysis, properties, and design of a Stewart-platform transducer, *J. Mech. Transmiss. Autom. Des.* **111**(1), 25–28 (1989)
- 12.74 C.C. Nguyen, S.S. Antrazi, Z.L. Zhou: Analysis and experimentation of a Stewart platform-based force/torque sensor, *Int. J. Robot. Autom.* **7**(3), 133–141 (1992)
- 12.75 C. Reboulet, A. Robert: Hybrid control of a manipulator with an active compliant wrist. In: *3rd ISRR*, Gouvieux, France, (1985) pp. 76–80
- 12.76 J. Duffy: *Statics and Kinematics with Applications to Robotics* (Cambridge University Press, New-York 1996)
- 12.77 C. Huang, W.-H. Hung, I. Kao: New conservative stiffness mapping for the Stewart-Gough platform. In: *IEEE Int. Conf. on Robotics and Automation*, Washington, (2002) pp. 823–828
- 12.78 J.L. Herder: *Energy-Free Systems: Theory, Conception and Design of Statically Balanced Spring Mechanisms*. Ph.D. Thesis, Delft University of Technology, Delft, (2001)
- 12.79 G.R. Dunlop, T.P. Jones: Gravity counter balancing of a parallel robot for antenna aiming. In: *6th ISRAM*, Montpellier, (1996) pp. 153–158
- 12.80 M. Jean, C. Gosselin: Static balancing of planar parallel manipulators. In: *IEEE Int. Conf. on Robotics and Automation*, Minneapolis, (1996) pp. 3732–3737
- 12.81 I. Ebert-Uphoff, C.M. Gosselin, T. Laliberté: Static balancing of spatial parallel platform-revisited, *ASME J. Mech. Des.* **122**(1), 43–51 (2000)
- 12.82 C.M. Gosselin, J. Wang: Static balancing of spatial six-degree-of-freedom parallel mechanisms with revolute actuators, *J. Robot. Syst.* **17**(3), 159–170 (2000)
- 12.83 M. Leblond, C.M. Gosselin: Static balancing of spatial and planar parallel manipulators with prismatic actuators. In: *ASME Design Engineering Technical Conferences*, Atlanta, (1998)
- 12.84 B. Monsarrat, C.M. Gosselin: Workspace analysis and optimal design of a 3-leg 6-DOF parallel platform mechanism, *IEEE Trans. Robot. Autom.* **19**(6), 954–966 (2003)

- 12.85 J. Wang, C.M. Gosselin: Static balancing of spatial three-degree-of-freedom parallel mechanisms, *Mechanism Machine Theory* **34**(3), 437–452 (1999)
- 12.86 Y. Wu, C.M. Gosselin: Synthesis of reactionless spatial 3-dof and 6-dof mechanisms without separate counter-rotations, *Int. J. Robot. Res.* **23**(6), 625–642 (2004)
- 12.87 M. Ait-Ahmed: Contribution à la modélisation géométrique et dynamique des robots parallèles. Ph.D. Thesis (Université Paul Sabatier, Toulouse 1993)
- 12.88 M.-J. Liu, C.-X. Li, C.-N. Li: Dynamics analysis of the Gough-Stewart platform manipulator, *IEEE Trans. Robot. Autom.* **16**(1), 94–98 (2000)
- 12.89 G.F. Liu, X.Z. Wu, Z.X. Li: Inertial equivalence principle and adaptive control of redundant parallel manipulators. In: *IEEE Int. Conf. on Robotics and Automation*, Washington, (2002) pp. 835–840
- 12.90 R. Clavel: Conception d'un robot parallèle rapide à 4 degrés de liberté. Ph.D. Thesis (EPFL, Lausanne 1991), n° 925
- 12.91 J. Gallardo, J.M. Rico, A. Frisoli, D. Checcacci, M. Bergamasco: Dynamics of parallel manipulators by means of screw theory, *Mechanism Machine Theory* **38**(11), 1113–1131 (2003)
- 12.92 L.-W. Tsai: Solving the inverse dynamics of a Stewart-Gough manipulator by the principle of virtual work, *ASME J. Mech. Des.* **122**(1), 3–9 (2000)
- 12.93 J. Wang, C.M. Gosselin: A new approach for the dynamic analysis of parallel manipulators, *Multibody Syst. Dyn.* **2**(3), 317–334 (1998)
- 12.94 Z. Geng, L.S. Haynes: On the dynamic model and kinematic analysis of a class of Stewart platforms, *Robot. Auton. Syst.* **9**(4), 237–254 (1992)
- 12.95 K. Liu, F. Lewis, G. Lebre, D. Taylor: The singularities and dynamics of a Stewart platform manipulator, *J. Intell. Robot. Syst.* **8**, 287–308 (1993)
- 12.96 K. Miller, R. Clavel: The Lagrange-based model of Delta-4 robot dynamics, *Robotersysteme* **8**(1), 49–54 (1992)
- 12.97 K. Miller: Optimal design and modeling of spatial parallel manipulators, *Int. J. Robot. Res.* **23**(2), 127–140 (2004)
- 12.98 A. Codourey, E. Burdet A body oriented method for finding a linear form of the dynamic equations of fully parallel robot. In: *IEEE Int. Conf. on Robotics and Automation*, Albuquerque, (1997) pp. 1612–1618
- 12.99 B. Dasgupta, P. Choudhury: A general strategy based on the Newton-Euler approach for the dynamic formulation of parallel manipulators, *Mechanism Machine Theory* **34**(6), 801–824 (1999)
- 12.100 P. Guglielmetti: Model-Based control of fast parallel robots: a global approach in operational space. Ph.D. Thesis (EPFL, Lausanne 1994)
- 12.101 K. Harib, K. Srinivasan: Kinematic and dynamic analysis of Stewart platform-based machine tool structures, *Robotica* **21**(5), 541–554 (2003)
- 12.102 W. Khalil, O. Ibrahim: General solution for the dynamic modeling of parallel robots. In: *IEEE Int. Conf. on Robotics and Automation*, New Orleans, (2004) pp. 3665–3670
- 12.103 C. Reboulet, T. Berthomieu: Dynamic model of a six degree of freedom parallel manipulator. In: *ICAR, Pise*, (1991) pp. 1153–1157
- 12.104 H. Abdellatif, B. Heimann: Adapted time-optimal trajectory planning for parallel with full dynamic modelling. In: *IEEE Int. Conf. on Robotics and Automation*, Barcelona, (2005) pp. 413–418
- 12.105 S. Tadokoro: Control of parallel mechanisms, *Adv. Robot.* **8**(6), 559–571 (1994)
- 12.106 M. Honegger, A. Codourey, E. Burdet: Adaptive control of the Hexaglide, a 6 dof parallel manipulator. In: *IEEE Int. Conf. on Robotics and Automation*, Albuquerque, (1997) pp. 543–548
- 12.107 S. Bhattacharya, H. Hatwal, A. Ghosh: An on-line estimation scheme for generalized Stewart platform type parallel manipulators, *Mechanism Machine Theory* **32**(1), 79–89 (1997)
- 12.108 J. Hesselbach, O. Becker, M. Krefft, I. Pietsch, N. Plitea: Dynamic modelling of plane parallel robot for control purposes. In: *3rd Chemnitz Parallelkinematik Seminar*, Chemnitz, (2002) pp. 391–409
- 12.109 P. Guglielmetti, R. Longchamp: A closed-form inverse dynamics model of the Delta parallel robot. In: *4th IFAC Symp. on Robot Control*, Syroco, Capri, (1994) pp. 51–56
- 12.110 K. Miller: Modeling of dynamics and model-based control of DELTA direct-drive parallel robot, *J. Robot. Mechatron.* **17**(4), 344–352 (1995)
- 12.111 E. Burdet, M. Honegger, A. Codourey: Controllers with desired dynamic compensation and their implementation on a 6 dof parallel manipulator. In: *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, Takamatsu, (2000)
- 12.112 K. Yamane, M. Okada, N. Komine, Y. Nakamura: Parallel dynamics computation and h_{∞} acceleration control of parallel manipulators for acceleration display, *J. Dyn. Syst. Meas. Control* **127**(2), 185–191 (2005)
- 12.113 J.E. McInroy: Modeling and design of flexure jointed Stewart platforms for control purposes, *IEEE/ASME Trans. Mechatron.* **7**(1), 95–99 (2002)
- 12.114 F. Xi: Dynamic balancing of hexapods for high-speed applications, *Robotica* **17**(3), 335–342 (1999)
- 12.115 J. Angeles The robust design of parallel manipulators. In: *1st Int. Colloquium, Collaborative Research Centre 562*, Braunschweig, (2002) pp. 9–30
- 12.116 S. Bhattacharya, H. Hatwal, A. Ghosh: On the optimum design of a Stewart platform type parallel manipulators, *Robotica* **13**(2), 133–140 (1995)

- 12.117 K.E. Zanganeh, J. Angeles: Kinematic isotropy and the optimum design of parallel manipulators, Int. J. Robot. Res. **16**(2), 185–197 (1997)
- 12.118 H. Fang, J.-P. Merlet: Multi-criteria optimal design of parallel manipulators based on interval analysis, Mechanism Machine Theory **40**(2), 151–171 (2005)
- 12.119 www-nads-sc.uiowa.edu
- 12.120 www.sop.inria.fr/coprin/equipe/merlet/merlet_eng.html
- 12.121 www.parallemic.org
- 12.122 J.-P. Merlet: *Parallel Robots* (Springer, Heidelberg 2006)
- 12.123 L.-W. Tsai: *Robot Analysis: The Mechanics of Serial and Parallel Manipulators* (Wiley, Hoboken 1999)

Robots with

13. Robots with Flexible Elements

Alessandro De Luca, Wayne Book

Design issues, dynamic modeling, trajectory planning, and feedback control problems are presented for robot manipulators having components with mechanical flexibility, either concentrated at the joints or distributed along the links. The chapter is divided accordingly in two main parts. Similarities or differences between the two types of flexibility are pointed out wherever appropriate.

For robots with flexible joints, the dynamic model is derived in detail by following a Lagrangian approach and possible simplified versions are discussed. The problem of computing the nominal torques that produce a desired robot motion is then solved. Regulation and trajectory tracking tasks are addressed by means of linear and nonlinear feedback control designs.

For robots with flexible links, relevant factors that lead to the consideration of distributed flexibility are analyzed. Dynamic models are presented, based on the treatment of flexibility through lumped elements, transfer matrices, or assumed modes. Several specific issues are then highlighted, including the selection of sensors,

13.1 Robots with Flexible Joints	288
13.1.1 Dynamic Modeling	289
13.1.2 Inverse Dynamics	293
13.1.3 Regulation Control	296
13.1.4 Trajectory Tracking	301
13.1.5 Further Reading	305
13.2 Robots with Flexible Links	306
13.2.1 Design Issues	306
13.2.2 Modeling of Flexible Link Arms	308
13.2.3 Control	312
13.2.4 Further Reading	316
References	317

the model order used for control design, and the generation of effective commands that reduce or eliminate residual vibrations in rest-to-rest maneuvers. Feedback control alternatives are finally discussed.

In each of the two parts of this chapter, a section is devoted to the illustration of the original references and to further readings on the subject.

The standard assumption underlying robot kinematics, dynamics, and control design is that manipulators consist only of rigid bodies (links and motion transmission components). This is, however, an ideal situation that may be considered valid only for slow motion and small interacting forces. In practice, mechanical flexibility in robot manipulators is present for two main reasons: the use of compliant transmission elements and the reduction of the mass of the moving links through the use of lightweight material and slender design. These two kinds of flexibility introduce static and dynamic deflections between the position of the driving actuators and the position of the manipulator end-effector. If flexibility is not taken into account when considering robot

design and control, a degradation of the overall expected performance of the robot typically occurs.

From a modeling point of view, flexibility can be assumed as concentrated at the robot joints or distributed (in different ways) along the robot links. The dynamic modeling steps are similar, with the need to introduce additional generalized coordinates besides those used to describe the rigid motion of the robot arm. However, the properties of the resulting models are quite different from a control point of view. Therefore, the case of *robots with flexible joints* and of *robots with flexible links* are presented in this chapter mostly as separate items, pointing out similarities or structural differences wherever appropriate. For both classes of

flexible robots, the relevant design issues, dynamic modeling, inverse dynamics algorithms, and control laws for set-point regulation and trajectory tracking will be discussed. Indeed, joint and link flexibilities may be present (and dynamically interact) at the same time. Many of

the presented results can be extended to this case as well.

In the following, it is assumed that the reader is already familiar with basic issues on kinematics, dynamics, and control of rigid robots (Chaps. 1, 2, 6).

13.1 Robots with Flexible Joints

The presence of joint flexibility is common in current industrial robots, when motion transmission/reduction elements such as belts – [as in the selective compliant assembly robot arm (SCARA) family], long shafts (as in the forearm of the Unimation Puma arm), cables, harmonic drives, or cycloidal gears are used. The purpose of these components is to allow relocation of the actuators next to the robot base, thus improving dynamic efficiency, or to guarantee high reduction ratios with power-efficient compact inline devices.

However, when subject to the forces/torques arising in normal robot operation, these components are intrinsically flexible (e.g., the flexspline in harmonic drives, see Fig. 13.1) and introduce a time-varying displacement between the position of the actuators and that of the driven links. Without a specific control action, an oscillatory behavior, typically of small magnitude but at a relatively high frequency, is observed at the robot end-effector level during free motion. In addition, some form of instability (e.g., chattering) may occur in tasks involving contact with the environment.

Recently, flexible actuation/transmission elements have been deliberately selected in robots intended for physical human–robot interaction. In fact, this form of mechanical compliance guarantees an inertial decoupling between the actuator and the (possibly, lightweight) link, thus reducing the kinetic energy

involved in undesired collisions with humans. Such a safety-oriented mechanical design should be traded-off with a more complex control design aimed at preserving the same performance of rigid robots in terms of speed and accuracy of the end-effector motion (Chap. 57).

Figures 13.2 and 13.3 show two examples of robot manipulators with flexible joints, the 8R robot *Dexter* and the 7R lightweight manipulator *DLR LWR-III*. The first robot has motors 3–8 located inside the second link,

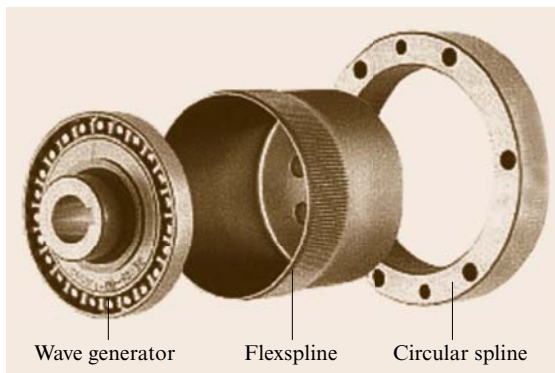


Fig. 13.1 The components of a harmonic drive



Fig. 13.2 The cable-driven robot *Dexter* by Scienza Machinale



Fig. 13.3 The lightweight manipulator *DLR LWR-III* by the German Aerospace Center

with motion transmitted to the distal links through steel cables and pulleys; the second has a modular structure, with each motor integrated in the associated joint and using a harmonic drive as reduction element. The joint stiffnesses of the *Dexter* robot vary between 120 and 6300 Nm/rad depending on the joint, while for the *DLR LWR-III* manipulator the range is 6000–15 000 Nm/rad. These numerical values refer to deflections and associated torques evaluated after the reduction gears.

Deflection of the flexible transmission elements can be modeled as being *concentrated at the joints* of the robot, thus limiting the complexity of the associated equations of motion. When compared to the rigid case, the dynamic model of robots with flexible joints (and rigid links) requires *twice* the number of generalized coordinates to completely characterize the configuration of all rigid bodies (motors and links) constituting the arm.

As a result, the case of flexible joint robots is one example in which the number of control inputs is strictly less than the number of mechanical degrees of freedom. This suggests that the design of control schemes

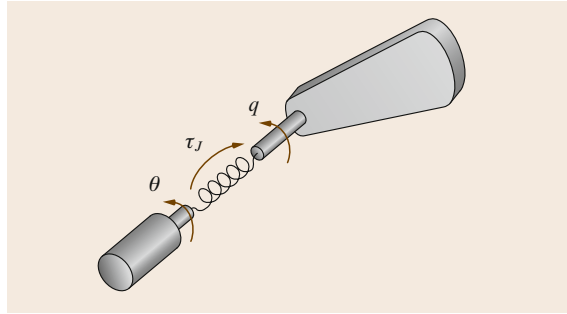


Fig. 13.4 Schematic representation of an elastic joint

realizing standard motion tasks is expected to be more difficult than for rigid robots. Moreover, the implementation of a full state feedback law will require twice the number of sensors, measuring quantities that are *before* and *after* (or across) the joint deformation. On the other hand, the motor torques used to command the robot as well as the disturbance torques due to joint flexibility perform work on the same joint axes (they are *physically collocated*). It should be noted that this is one relevant difference with respect to the case of flexibility distributed along the links. This characteristic is very helpful for rejecting vibrations and controlling the overall robot motion. Roughly speaking, we will be able to use input commands acting *before* the source of flexibility to ensure that output variables defined beyond the flexibility behave in a desired way.

13.1.1 Dynamic Modeling

Dynamic models that explicitly include joint flexibility are used to evaluate quantitatively vibratory effects superposed on the rigid motion, verify whether the control laws obtained under the assumption of joint rigidity can still work in practice (or to what extent they should be modified), and enable the design of new model-based feedforward and feedback controllers.

We consider a robot with flexible joints as an open kinematic chain having $N + 1$ rigid bodies, the base and the N links, interconnected by N (rotary or prismatic) joints undergoing deflection, and actuated by N electrical drives. From a mechanical point of view, each motor (with its stator and rotor) is an additional rigid body with its inertial properties. All joints are considered to be flexible, though mixed situations may be encountered due to the use of different transmission devices. When reduction gearings are present, they are modeled as being placed *before* the joint deflection occurs.

The following standard assumptions are made.

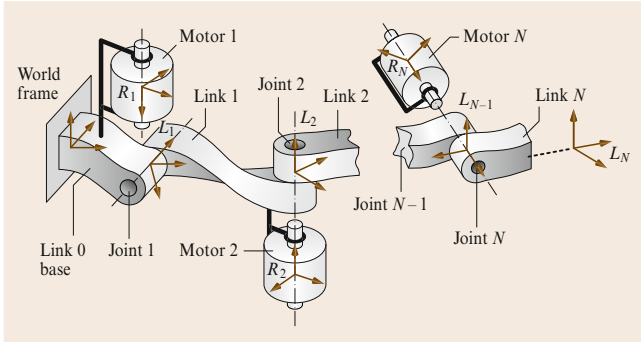


Fig. 13.5 Arrangement of motors and links in an open kinematic chain

- A1** Joint deflections are small, so that flexibility effects are limited to the domain of linear elasticity.
- A2** The actuators' rotors are modeled as uniform bodies having their center of mass on the rotation axis.
- A3** Each motor is located on the robot arm in a position preceding the driven link. (This can be generalized to the case of multiple motors simultaneously driving multiple distal links.)

The first assumption supports the terminology of robots with *elastic* joints often used in the literature. The elasticity at joint i is modeled by a spring of stiffness $K_i > 0$, which is torsional for rotational joints and linear for translational ones. Figure 13.4 shows a single link driven by a motor through a rotational elastic joint. Nonlinear stiffness characteristics for the flexible joint can also be considered, provided that the map from deflection to force is smooth and invertible. Assumption **A2** is a basic requirement for long life of an electrical drive and thus very reasonable. As we shall see, it implies that the inertia matrix and the gravity term in the robot dynamic model will be independent of the angular position of the motors. A typical arrangement of motors satisfying assumption **A3** is shown in Fig. 13.5. The simplest situation is when the i -th motor moves link i and is mounted on link $i - 1$ with its rotation axis aligned with the i -th joint. This is, for instance, the case for the *LWR-III* manipulator. The dislocation of the actuating motors along the structure has a great influence on the structure of the dynamic equations of motion.

For kinematic and dynamic analysis, $2N$ frames are attached to the $2N$ moving rigid bodies (links and motors) in the robot chain: the link frames L_i and the motor frames R_i , for $i = 1, \dots, N$ (Fig. 13.5). For the

definition of the link frames L_i , the standard Denavit–Hartenberg convention can be used. The frames R_i are attached to the motor stators, aligned with the axes of symmetry of the motors and with the z -axis along the spinning direction of the rotor.

Accordingly, $2N$ generalized coordinates will be needed. A possible set of coordinates is given by

$$\Theta = \begin{pmatrix} q \\ \theta \end{pmatrix} \in \mathbb{R}^{2N},$$

where q is the N -vector of link positions and θ is the N -vector of motor (i.e., rotor) positions, as reflected through the transmission/reduction gears. This choice of variables is quite convenient because:

1. the model will be formally independent of the reduction ratios;
2. these position variables will have a similar dynamic range;
3. the kinematics of the robot will be a function of the link variables q only (these variables are already *beyond* the joint flexibility) so that all issues related to direct/inverse kinematics will be identical to the case of fully rigid robots.

For some considerations, it is useful to define also the variable θ_m , namely the N -vector of motor positions before reduction, which are the quantities directly measured by encoders mounted on the motors. In the case of a motor directly placed on the i -th joint axis, one has $\dot{\theta}_{m,i} = n_i \dot{\theta}_i$, where $n_i \geq 1$ is the reduction ratio at the i -th joint. In addition, for $i = 1, \dots, N$, the difference $\delta_i = q_i - \theta_i$ is the deflection at the i -th joint, while $\tau_{J,i} = K_i(\theta_i - q_i)$ is the torque transmitted to the i -th link through the associated elastic spring (Fig. 13.4) – the quantity measured by a joint torque sensor, whenever present. Note that for robots with flexible links, the set (θ, δ) is typically used in the dynamic modeling, where δ is the vector of link deflection coordinates.

Following a Lagrangian approach, the single energy contributions to the Lagrangian $\mathcal{L} = \mathcal{T}(\Theta, \dot{\Theta}) - \mathcal{U}(\Theta)$ will be derived next.

The *potential energy* of the robot is due to gravity and joint elasticity. The gravity part is related to the position of the barycenter of the links (each of mass m_i) and of the motors (of mass m_{r_i}). Because of assumption **A2**, the latter will be independent of θ . Thus,

$$\mathcal{U}_{\text{grav}} = \mathcal{U}_{\text{grav,link}}(q) + \mathcal{U}_{\text{grav,motor}}(q).$$

For the elastic part of the potential, because of assumption **A1**, we have

$$\mathcal{U}_{\text{elas}} = \frac{1}{2} (\mathbf{q} - \boldsymbol{\theta})^\top \mathbf{K} (\mathbf{q} - \boldsymbol{\theta}),$$

$$\mathbf{K} = \text{diag}(K_1, \dots, K_N).$$

As a result, $\mathcal{U}(\boldsymbol{\Theta}) = \mathcal{U}_{\text{grav}}(\mathbf{q}) + \mathcal{U}_{\text{elas}}(\mathbf{q} - \boldsymbol{\theta})$.

The *kinetic energy* of the robot is the sum of the link and rotor contributions. For the links, there is no difference with respect to the standard rigid robot case and it will be sufficient to write in general

$$\mathcal{T}_{\text{link}} = \frac{1}{2} \dot{\mathbf{q}}^\top \mathbf{M}_L(\mathbf{q}) \dot{\mathbf{q}}, \quad (13.1)$$

with the positive-definite symmetric link inertia matrix $\mathbf{M}_L(\mathbf{q})$. For the rotors, some more detail is needed:

$$\begin{aligned} \mathcal{T}_{\text{rotor}} &= \sum_{i=1}^N \mathcal{T}_{\text{rotor}_i} \\ &= \sum_{i=1}^N \left(\frac{1}{2} m_{r_i} \mathbf{v}_{r_i}^\top \mathbf{v}_{r_i} + \frac{1}{2} {}^{R_i} \boldsymbol{\omega}_{r_i}^\top \mathbf{I}_{r_i} {}^{R_i} \boldsymbol{\omega}_{r_i} \right), \end{aligned} \quad (13.2)$$

where \mathbf{v}_{r_i} is the linear velocity of the center of mass of the i -th rotor and $\boldsymbol{\omega}_{r_i}$ is the angular velocity of the i -th rotor body. All quantities in the angular contributions in (13.2) are conveniently expressed in the local frame R_i . According to assumption **A2**, the rotor inertia matrix is then diagonal

$${}^{R_i} \mathbf{I}_{r_i} = \text{diag}(I_{r_{i,xx}}, I_{r_{i,yy}}, I_{r_{i,zz}}),$$

with $I_{r_{i,xx}} = I_{r_{i,yy}}$, while \mathbf{v}_{r_i} can be expressed as a function of \mathbf{q} and $\dot{\mathbf{q}}$ only. Moreover, due to assumption **A3**, the angular velocity of the i -th rotor has the general expression

$${}^{R_i} \boldsymbol{\omega}_{r_i} = \sum_{j=1}^{i-1} \mathbf{J}_{r_i,j}(\mathbf{q}) \dot{q}_j + \begin{pmatrix} 0 \\ 0 \\ \dot{\theta}_{m,i} \end{pmatrix}, \quad (13.3)$$

where $\mathbf{J}_{r_i,j}(\mathbf{q})$ is the j -th column of the Jacobian relating the link velocities $\dot{\mathbf{q}}$ to the angular velocity of the i -th rotor in the robot chain. By substituting (13.3) into (13.2) and expressing $\dot{\theta}_m$ in terms of $\dot{\boldsymbol{\theta}}$, it can be shown that

$$\begin{aligned} \mathcal{T}_{\text{rotor}} &= \frac{1}{2} \dot{\mathbf{q}}^\top \left[\mathbf{M}_R(\mathbf{q}) + \mathbf{S}(\mathbf{q}) \mathbf{B}^{-1} \mathbf{S}^\top(\mathbf{q}) \right] \dot{\mathbf{q}} \\ &\quad + \dot{\mathbf{q}}^\top \mathbf{S}(\mathbf{q}) \dot{\boldsymbol{\theta}} + \frac{1}{2} \dot{\boldsymbol{\theta}}^\top \mathbf{B} \dot{\boldsymbol{\theta}}, \end{aligned} \quad (13.4)$$

where \mathbf{B} is the constant diagonal inertia matrix collecting the rotors inertial components $I_{r_{i,zz}}$ around their spinning axes, $\mathbf{M}_R(\mathbf{q})$ contains the rotor masses (and, possibly,

the rotor inertial components along the other principal axes), and the square matrix $\mathbf{S}(\mathbf{q})$ expresses the inertial couplings between the rotors and the previous links in the robot chain.

A simple example illustrates the derivation and the actual expressions of the terms in (13.4). We also include the reduction elements to show how the rotor inertial component around the spinning axis will appear in the kinetic energy, weighted by different powers of the reduction ratios. Consider a planar robot with two rotational elastic joints, a first link of length ℓ_1 , and motors mounted directly on the joint axes. The kinetic energies of the two rotors are

$$\begin{aligned} \mathcal{T}_{\text{rotor}_1} &= \frac{1}{2} I_{r_{1,zz}} \dot{\theta}_{m,1}^2 = \frac{1}{2} I_{r_{1,zz}} n_1^2 \dot{\theta}_1^2 \\ \mathcal{T}_{\text{rotor}_2} &= \frac{1}{2} m_{r_2} \ell_1^2 \dot{q}_1^2 + \frac{1}{2} I_{r_{2,zz}} (\dot{q}_1 + \dot{\theta}_{m,2})^2 \\ &= \frac{1}{2} m_{r_2} \ell_1^2 \dot{q}_1^2 + \frac{1}{2} I_{r_{2,zz}} (\dot{q}_1^2 + 2n_2 \dot{q}_1 \dot{\theta}_2 + n_2^2 \dot{\theta}_2^2), \end{aligned}$$

leading to

$$\begin{aligned} \mathbf{B} &= \begin{pmatrix} I_{r_{1,zz}} n_1^2 & 0 \\ 0 & I_{r_{2,zz}} n_2^2 \end{pmatrix}, & \mathbf{S} &= \begin{pmatrix} 0 & I_{r_{2,zz}} n_2 \\ 0 & 0 \end{pmatrix}, \\ \mathbf{M}_R &= \begin{pmatrix} m_{r_2} \ell_1^2 & 0 \\ 0 & 0 \end{pmatrix}, & \mathbf{S} \mathbf{B}^{-1} \mathbf{S}^\top &= \begin{pmatrix} I_{r_{2,zz}} & 0 \\ 0 & 0 \end{pmatrix}. \end{aligned}$$

In this case, the matrix \mathbf{S} (as well as \mathbf{M}_R) is constant. Note that for large reduction ratios n_i , the dominant inertial effect due to the rotors is given by the matrix \mathbf{B} . Also, if the second motor were mounted remotely on the first joint (as it is often the case in the class of **SCARA** arms), or still close to the second joint but with the spinning axis orthogonal to the axis of this joint, then the matrix \mathbf{S} would be zero.

In general, as a consequence of assumption **A3**, the matrix $\mathbf{S}(\mathbf{q})$ always has a strictly upper-triangular structure with a cascaded dependence of its nonzero elements:

$$\mathbf{S}(\mathbf{q}) = \begin{pmatrix} 0 & S_{12} S_{13}(q_2) S_{14}(q_2, q_3) \dots & \dots & S_{1N}(q_2, \dots, q_{N-1}) \\ 0 & 0 & S_{23} & S_{24}(q_3) & \dots & \dots & S_{2N}(q_3, \dots, q_{N-1}) \\ 0 & 0 & 0 & S_{34} & \dots & \dots & S_{3N}(q_4, \dots, q_{N-1}) \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & S_{N-2,N-1} S_{N-2,N}(q_{N-1}) \\ 0 & 0 & 0 & \dots & 0 & 0 & S_{N-1,N} \\ 0 & 0 & 0 & \dots & 0 & 0 & 0 \end{pmatrix}. \quad (13.5)$$

Summing up, the total kinetic energy of the robot is

$$\begin{aligned}\mathcal{T} &= \frac{1}{2} \dot{\Theta}^\top \mathcal{M}(\Theta) \dot{\Theta} \\ &= \frac{1}{2} \begin{pmatrix} \dot{q}^\top & \dot{\theta}^\top \end{pmatrix} \begin{pmatrix} \mathcal{M}(q) & S(q) \\ S^\top(q) & B \end{pmatrix} \begin{pmatrix} \dot{q} \\ \dot{\theta} \end{pmatrix},\end{aligned}$$

with

$$\mathcal{M}(q) = M_L(q) + M_R(q) + S(q)B^{-1}S^\top(q). \quad (13.6)$$

As anticipated, the total inertia matrix \mathcal{M} of the robot depends only on q .

Using the Lagrange equations finally yields the complete dynamic model

$$\begin{aligned}\begin{pmatrix} \mathcal{M}(q)S(q) \\ S^\top(q) B \end{pmatrix} \begin{pmatrix} \ddot{q} \\ \ddot{\theta} \end{pmatrix} + \begin{pmatrix} c(q, \dot{q}) + c_1(q, \dot{q}, \dot{\theta}) \\ c_2(q, \dot{q}) \end{pmatrix} \\ + \begin{pmatrix} g(q) + K(q - \theta) \\ K(\theta - q) \end{pmatrix} = \begin{pmatrix} \theta \\ \tau \end{pmatrix},\end{aligned} \quad (13.7)$$

where the inertial terms (related to the total inertia matrix $\mathcal{M}(q)$), the Coriolis and centrifugal terms (collectively denoted by $c_{\text{tot}}(\Theta, \dot{\Theta})$), and the potential terms ($\partial \mathcal{U}(\Theta)/\partial \Theta$) have been written separately. In particular, $g(q) = (\partial \mathcal{U}_{\text{grav}}(q)/\partial q)^\top$ while $\tau_J = K(\theta - q)$ is the elastic torque transmitted through the joints.

The first N and the last N equations of the dynamic model (13.7) are referred to as the *link* and the *motor equations*, respectively.

On the right-hand side of (13.7), all nonconservative generalized forces should appear. When dissipative effects are not considered, only the motor torque τ producing work on the θ variable is present (i. e., the torques on the motor output shafts, amplified by the square of the reduction ratios) in the motor equations. If the robot end-effector is in contact with the environment, the zero in the right-hand side of the link equations should be replaced by $\tau_{\text{ext}} = J^\top(q)F$, where $J(q)$ is the robot Jacobian and F the forces/torques acting from the environment on the robot.

In the presence of energy-dissipating effects, additional terms appear in the right-hand side of (13.7). For instance, viscous friction at both sides of the transmissions and spring damping of the (visco)elastic joints give rise to the vector term

$$\begin{pmatrix} -F_q \dot{q} - D(\dot{q} - \dot{\theta}) \\ -F_\theta \dot{\theta} - D(\dot{\theta} - \dot{q}) \end{pmatrix}, \quad (13.8)$$

where the diagonal, positive-definite matrices F_q , F_θ , and D contain, respectively, the viscous coefficients on

the link side and on the motor side, and the damping of the elastic springs at the joints. More general forms of nonlinear friction τ_F can be considered. Note that, in principle, friction acting on the motor side can be fully compensated by a suitable choice of the control torque τ , while this is not true for friction acting on the link side, due to the noncollocation.

Model Properties

All elements in the $2N$ -vector $c_{\text{tot}}(\Theta, \dot{\Theta})$ of velocity-dependent terms in (13.7) are independent of the motor position θ . The specific dependence of the N -vectors c , c_1 , and c_2 follows from the general expression of the components of c_{tot} based on Christoffel symbols:

$$c_{\text{tot},i}(\Theta, \dot{\Theta}) = \frac{1}{2} \dot{\Theta}^\top \left[\frac{\partial \mathcal{M}_i}{\partial \Theta} + \left(\frac{\partial \mathcal{M}_i}{\partial \Theta} \right)^\top - \frac{\partial \mathcal{M}}{\partial \Theta_i} \right] \dot{\Theta},$$

for $i = 1, \dots, 2N$, where \mathcal{M}_i is the i -th column of the total inertia matrix $\mathcal{M}(\Theta)$. In particular, the velocity vectors c_1 and c_2 contain terms arising from the presence of the $S(q)$ matrix. Performing calculations, it can be shown that:

1. the vector c_1 does not contain quadratic velocity terms in \dot{q} or $\dot{\theta}$, but only in $\dot{\theta}_i \dot{q}_j$;
2. when the matrix S is constant, both c_1 and c_2 vanish.

The dynamic model (13.7) also shares some properties of the rigid case, such as:

- the model equations admit a linear parametrization in terms of a suitable set of dynamic coefficients, including joint stiffnesses and motor inertias, which is useful for model identification and adaptive control;
- the Coriolis and centrifugal terms can always be factorized as $c_{\text{tot}}(\Theta, \dot{\Theta}) = \mathcal{C}(\Theta, \dot{\Theta})\dot{\Theta}$, in such a way that matrix $\mathcal{M} - 2\mathcal{C}$ is skew-symmetric – a property used in control analysis;
- for robots having only rotational joints, the gradient of the gravity vector $g(q)$ is globally bounded in norm by a constant.

Finally, when the joint stiffness is extremely large ($K \rightarrow \infty$), then $\theta \rightarrow q$ while $\tau_J \rightarrow \tau$. It is easy to check that the dynamic model (13.7) collapses in this limit to the standard model of fully rigid robots (including links and motors).

Reduced Model

In general, the link and motor equations in (13.7) are not only dynamically coupled through the elastic torque τ_J

at the joints, but also (at the acceleration level) via the inertial components of matrix $S(q)$ – usually a path of low energy transfer. The presence and actual relevance of these inertial couplings depend on the kinematic arrangement of the manipulator arm and, in particular, on the specific location of the motors and transmission devices. There are cases in which the matrix S is constant (e.g., the planar case, as in the previous example, with any number of links) or zero (e.g., for a single link with elastic joint or for a robot with $N = 2$ links having the two joint axes orthogonal and the motors mounted at the joints). As a result, the dynamic equations will simplify considerably.

For a generic robot with elastic joints, one can take advantage of the presence of large reduction ratios (with the n_i of the order 100–150) and simply neglect energy contributions due to the inertial couplings between the motors and the links (see again the 2R planar example). This is equivalent to considering the following simplifying assumption:

A4 The angular velocity of the rotors is due only to their own spinning, i. e.,

$${}^{R_i}\omega_{r_i} = \begin{pmatrix} 0 & 0 & \dot{\theta}_{m,i} \end{pmatrix}^\top, \quad i = 1, \dots, N,$$

instead of the complete (13.3).

As a result, the total angular kinetic energy of the rotors is just $\frac{1}{2} \dot{\theta}^\top B \dot{\theta}$ (or $S \equiv 0$), and the dynamic model (13.7) reduces to

$$\begin{aligned} M(q)\ddot{q} + c(q, \dot{q}) + g(q) + K(q - \theta) &= 0 \\ B\ddot{\theta} + K(\theta - q) &= \tau, \end{aligned} \quad (13.9)$$

with $M(q) = M_L(q) + M_R(q)$. The main feature of this model is that the link and motor equations are dynamically coupled only through the elastic torque τ_J . In addition, the motor equations are now fully linear.

We note that the complete model (13.7) and the reduced model (13.9) display different characteristics with respect to certain control problems. In fact, the reduced model is always feedback linearizable by *static* state feedback, whereas this is never the case for the complete model as soon as a coupling $S \neq 0$ is present.

Singular Perturbation Model

It is interesting to show the *two-time-scale* dynamic behavior that it is induced in robots with elastic joints when the joint stiffness K is relatively large, but still finite. This behavior can be made explicit by a simple linear change of coordinates, namely replacing θ with the joint

torque τ_J . For the sake of simplicity, this is illustrated on the reduced model only (without dissipation).

Since the diagonal joint stiffness matrix is assumed to have large but similar elements, a common scalar factor $1/\epsilon^2 \gg 1$ can be extracted as

$$K = \frac{1}{\epsilon^2} \hat{K} = \frac{1}{\epsilon^2} \text{diag}(\hat{K}_1, \dots, \hat{K}_N).$$

The *slow* subsystem is then given by the link equations, once they are rewritten as

$$M(q)\ddot{q} + c(q, \dot{q}) + g(q) = \tau_J. \quad (13.10)$$

To obtain the dynamics of the *fast* subsystem, the joint torque is differentiated twice, the motor and link accelerations are replaced from (13.9), and the above definition of \hat{K} is used. This leads to

$$\begin{aligned} \epsilon^2 \ddot{\tau}_J &= \hat{K} \{ B^{-1} \tau - [B^{-1} + M^{-1}(q)] \tau_J \\ &\quad + M^{-1}(q)[c(q, \dot{q}) + g(q)] \}. \end{aligned} \quad (13.11)$$

For small ϵ , (13.10, 13.11) represent a singularly perturbed system. The two separate time scales governing the slow and fast dynamics are t and $\sigma = t/\epsilon$, since

$$\epsilon^2 \ddot{\tau}_J = \epsilon^2 \frac{d^2 \tau_J}{dt^2} = \frac{d^2 \tau_J}{d\sigma^2}.$$

This model serves as a basis for composite control schemes, where the control torque has the general form

$$\tau = \tau_s(q, \dot{q}, t) + \epsilon \tau_f(q, \dot{q}, \tau_J, \dot{\tau}_J). \quad (13.12)$$

This includes a slow action τ_s , designed when neglecting joint elasticity, and an additional action τ_f for locally stabilizing the fast flexible dynamics around a suitable manifold in the state space. It can be verified that, when setting $\epsilon = 0$ in (13.10, 11, 12), the equivalent rigid robot model is recovered as

$$[M(q) + B]\ddot{q} + c(q, \dot{q}) + g(q) = \tau_s.$$

A similar singular perturbation model (and control design) can also be derived for robot manipulators with flexible links.

13.1.2 Inverse Dynamics

Given a desired motion of a robot, we wish to compute the nominal torque needed to reproduce this motion exactly in ideal conditions (the inverse dynamics problem). Such nominal torque can be used as the feedforward term in a trajectory-tracking control law.

For rigid robots, the inverse dynamics is a straightforward algebraic computation obtained by replacing

the desired motion of the generalized coordinates in the dynamic model. The minimum requirement for exact reproduction is that the planned motion has a continuously differentiable desired velocity. For robots with elastic joints, a motion task can be conveniently expressed in terms of a desired link trajectory $\mathbf{q} = \mathbf{q}_d(t)$ (possibly obtained from the kinematic inversion of a desired motion in Cartesian space). The additional complexity lies in the fact that not all robot coordinates are directly assigned in this way, so that additional derivations should be performed. This will require some higher degree of smoothness of the desired trajectory $\mathbf{q}_d(t) \in [0, T]$, where the final time T may or may not be finite.

Reduced Model

Consider first the case of the reduced model (13.7) and set $\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q})$ for compactness. By evaluating the link equations on the desired link motion

$$\mathbf{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \mathbf{K}\mathbf{q}_d = \mathbf{K}\boldsymbol{\theta}_d, \quad (13.13)$$

the nominal position $\boldsymbol{\theta}_d$ of the motors associated with the desired link motion is readily obtained. The nominal elastic torque at the joints is $\boldsymbol{\tau}_{l,d} = \mathbf{K}(\boldsymbol{\theta}_d - \mathbf{q}_d)$; note that, from (13.13), this quantity can be expressed as a function of \mathbf{q}_d , $\dot{\mathbf{q}}_d$, and $\ddot{\mathbf{q}}_d$, which is independent of \mathbf{K} . Differentiating (13.13) leads to the expression for the nominal motor velocity $\dot{\boldsymbol{\theta}}_d$,

$$\mathbf{M}(\mathbf{q}_d)\mathbf{q}_d^{[3]} + \dot{\mathbf{M}}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \dot{\mathbf{n}}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \mathbf{K}\dot{\mathbf{q}}_d = \mathbf{K}\dot{\boldsymbol{\theta}}_d, \quad (13.14)$$

where the notation $\mathbf{y}^{[i]} = d^i \mathbf{y} / dt^i$ is used. Differentiating once more, we obtain

$$\begin{aligned} \mathbf{M}(\mathbf{q}_d)\mathbf{q}_d^{[4]} + 2\dot{\mathbf{M}}(\mathbf{q}_d)\mathbf{q}_d^{[3]} + \ddot{\mathbf{n}}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \\ + [\dot{\mathbf{M}}(\mathbf{q}_d) + \mathbf{K}]\ddot{\mathbf{q}}_d = \mathbf{K}\ddot{\boldsymbol{\theta}}_d, \end{aligned} \quad (13.15)$$

to be used in the motor equations evaluated along the desired motion. After simplifications, the nominal torque is obtained as

$$\begin{aligned} \boldsymbol{\tau}_d = & [\mathbf{M}(\mathbf{q}_d) + \mathbf{B}]\ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \\ & + \mathbf{B}\mathbf{K}^{-1} \left\{ \mathbf{M}(\mathbf{q}_d)\mathbf{q}_d^{[4]} + 2\dot{\mathbf{M}}(\mathbf{q}_d)\mathbf{q}_d^{[3]} \right. \\ & \left. + \ddot{\mathbf{n}}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \ddot{\mathbf{M}}(\mathbf{q}_d) + \mathbf{K}\ddot{\mathbf{q}}_d \right\}, \end{aligned} \quad (13.16)$$

where the extra contributions to the nominal torque of the rigid case due to joint elasticity can be clearly recognized. The evaluation of $\boldsymbol{\tau}_d$ involves the computation of

first- and second-order partial derivatives of the dynamic model terms. For instance, one needs to compute

$$\dot{\mathbf{M}}[\mathbf{q}_d(t)] = \sum_{i=1}^N \frac{\partial \mathbf{M}_i(\mathbf{q})}{\partial \mathbf{q}} \Big|_{\mathbf{q}=\mathbf{q}_d(t)} \dot{\mathbf{q}}_d(t) \mathbf{e}_i^\top,$$

where \mathbf{e}_i is the i -th unit vector. This and other similar expressions can be obtained by symbolic manipulation software. From (13.16) it follows that the minimum requirement for the exact reproducibility of the desired motion is that $\mathbf{q}_d(t)$ admits a continuously differentiable jerk (i.e., that $\mathbf{q}_d^{[4]}(t)$ exists in the time interval $[0, T]$). Such a smoother requirement should not come unexpected, in view of the flexible nature of the system.

Complete Model

Some more analysis is needed for the model (13.9). For ease of exposition, and without loss of generality, the case of a constant matrix \mathbf{S} is considered. When evaluating the link equations on the desired link motion

$$\mathbf{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{S}\ddot{\boldsymbol{\theta}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \mathbf{K}\mathbf{q}_d = \mathbf{K}\boldsymbol{\theta}_d, \quad (13.17)$$

the additional presence of the motor acceleration in the left-hand side does not allow the motor position $\boldsymbol{\theta}_d$ to be expressed directly as a function of $(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)$ only. However, the strictly upper-triangular structure (13.5) of \mathbf{S} allows one to define $\boldsymbol{\theta}_d$ componentwise, using the scalar equations in (13.17) recursively. The N -th equation is in fact independent of $\ddot{\boldsymbol{\theta}}_d$,

$$\mathbf{M}_N^\top(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{0}^\top \ddot{\boldsymbol{\theta}}_d + n_N(\mathbf{q}_d, \dot{\mathbf{q}}_d) + K_N q_{d,N} = K_N \theta_{d,N},$$

so that this equation can be used to define

$$\theta_{d,N} = f_N(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)$$

and, after double differentiation, its second time derivative

$$\ddot{\theta}_{d,N} = f_N''(\mathbf{q}_d, \dot{\mathbf{q}}_d, \dots, \mathbf{q}_d^{[4]}).$$

In the $(N-1)$ -th equation,

$$\begin{aligned} \mathbf{M}_{N-1}^\top(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{S}_{N-1,N}\ddot{\boldsymbol{\theta}}_{d,N} \\ + n_{N-1}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + K_{N-1}q_{d,N-1} = K_{N-1}\theta_{d,N-1}, \end{aligned}$$

the acceleration $\ddot{\theta}_{d,N}$ has already been determined in the previous step. Therefore, this equation can be used similarly to define

$$\theta_{d,N-1} = f_{N-1}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \dots, \mathbf{q}_d^{[4]})$$

and, after two time differentiations, also

$$\ddot{\theta}_{d,N-1} = f_{N-1}''(\mathbf{q}_d, \dot{\mathbf{q}}_d, \dots, \mathbf{q}_d^{[6]}).$$

Note that whenever $S_{N-1,N} = 0$, there will be no increase in the degree of the highest-order derivatives of \mathbf{q}_d within the functional dependence of $\theta_{d,N-1}$. This argument also applies recursively to the following steps. Proceeding backward through the link equations, the scalar computations end up with the definition of

$$\theta_{d,1} = f_1(\mathbf{q}_d, \dot{\mathbf{q}}_d, \dots, \mathbf{q}_d^{[2N]})$$

and

$$\ddot{\theta}_{d,1} = f_1''(\mathbf{q}_d, \dot{\mathbf{q}}_d, \dots, \mathbf{q}_d^{[2(N+1)]}),$$

where the dependence on the highest possible differential degree of \mathbf{q}_d is shown. With $\ddot{\theta}_d = f''(\cdot)$ made available in this way, the nominal torque is finally computed from the motor equations, again evaluated along the desired motion. Using (13.17) to substitute for $\mathbf{K}(\theta_d - \mathbf{q}_d)$ yields:

$$\begin{aligned} \tau_d = & \left[\mathbf{M}(\mathbf{q}_d) + \mathbf{S}^\top \right] \ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \\ & + (\mathbf{B} + \mathbf{S}) \ddot{\theta}_d(\mathbf{q}_d, \dot{\mathbf{q}}_d, \dots, \mathbf{q}_d^{[2(N+1)]}). \end{aligned} \quad (13.18)$$

As a result, the presence of inertial motor–link couplings considerably increases the complexity of the solution to the inverse dynamics problem. The exact reproduction of a desired link trajectory by the nominal torque in (13.18) requires that $\mathbf{q}_d(t)$ has a continuously differentiable $(2N+1)$ -th time derivative (i. e., that $\mathbf{q}_d^{[2(N+1)]}(t)$ exists in the time interval $[0, T]$). For rest-to-rest motions, this implies a trajectory plan that imposes a very slow start and ending phases.

We finally remark that the possibility of expressing the evolution of the state and input of a system algebraically in terms of the evolution of an output variable (in our case \mathbf{q}) and a finite number of its derivatives is sometimes referred to as the *flatness* property. The above inverse dynamics analysis shows that \mathbf{q} is a flat output for robots with elastic joints modeled either by (13.7) or by (13.9).

Note also that, when a *constant* \mathbf{q}_d is considered, these computations all provide the same condition for the associated motor position

$$\theta_d = \mathbf{q}_d + \mathbf{K}^{-1} \mathbf{g}(\mathbf{q}_d) \quad (13.19)$$

and nominal static torque

$$\tau_d = \mathbf{g}(\mathbf{q}_d). \quad (13.20)$$

Presence of Dissipative Terms

Inclusion of dissipative terms in the inverse dynamics deserves some additional comments. Any model of frictional effects acting on the motor side of the transmissions can be included in the computation without extra

requirements. Friction at the link side should instead be described by a smooth model, because of the need to differentiate the link equations. Thus, some functional approximations may be needed (e.g., replacing discontinuous *sign* functions with hyperbolic tangents) before including this term in (13.13) or (13.17).

On the other hand, the presence of a nonnegligible spring damping \mathbf{D} (13.8) changes the structure of the computations. While this reduces the order of the derivatives of \mathbf{q}_d involved, the problem itself will become nonalgebraic; in fact, the inverse system will require the use of the solution of a dynamical system, albeit a simple one.

Consider for instance the model (13.9), including now all the dissipative terms given in (13.8). When evaluating the link equations,

$$\begin{aligned} \mathbf{M}(\mathbf{q}_d) \ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + (\mathbf{D} + \mathbf{F}_q) \dot{\mathbf{q}}_d + \mathbf{K} \mathbf{q}_d \\ = \mathbf{D} \dot{\theta}_d + \mathbf{K} \theta_d, \end{aligned} \quad (13.21)$$

the motor velocity $\dot{\theta}_d$ additionally appears on the right-hand side. Differentiating (13.21) leads to

$$\mathbf{D} \ddot{\theta}_d + \mathbf{K} \dot{\theta}_d = \mathbf{w}_d, \quad (13.22)$$

with

$$\begin{aligned} \mathbf{w}_d = & \mathbf{M}(\mathbf{q}_d) \mathbf{q}_d^{[3]} + [\dot{\mathbf{M}}(\mathbf{q}_d) + \mathbf{D} + \mathbf{F}_q] \dot{\mathbf{q}}_d \\ & + \dot{\mathbf{n}}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \mathbf{K} \dot{\mathbf{q}}_d. \end{aligned}$$

Equation (13.22) represents a first-order linear *asymptotically stable* dynamical system (internal dynamics), with state $\dot{\theta}_d$ and forcing signal $\mathbf{w}_d(t)$. For a given $\dot{\theta}_d(0)$, its solution $\dot{\theta}_d(t)$ and the associated solution derivative $\ddot{\theta}_d(t)$ are needed to evaluate the nominal torque in the motor equations. This yields

$$\tau_d = \mathbf{M}(\mathbf{q}_d) \ddot{\mathbf{q}}_d + \mathbf{n}(\mathbf{q}_d, \dot{\mathbf{q}}_d) + \mathbf{F}_q \dot{\mathbf{q}}_d + \mathbf{B} \ddot{\theta}_d + \mathbf{F}_\theta \dot{\theta}_d,$$

where (13.21) has been used to replace the term $\mathbf{D}(\dot{\theta}_d - \dot{\mathbf{q}}_d) + \mathbf{K}(\theta_d - \mathbf{q}_d)$. In this case, the desired link trajectory $\mathbf{q}_d(t)$ should have a continuously differentiable acceleration ($\mathbf{q}_d^{[3]}$ should exist, since it is used in the definition of \mathbf{w}_d). Note that any initialization $\dot{\theta}_d(0)$ of the internal dynamics (13.22) is feasible (i. e., it produces a specific torque profile $\tau_d(t)$ yielding the same link motion $\mathbf{q}_d(t)$), with the associated motor position $\theta_d(0)$ initialized from (13.21). However, we should match the actual initial state of the robot arm. For instance, starting from an equilibrium state implies the unique choice $\dot{\theta}_d(0) = \mathbf{0}$.

A similar procedure can also be applied to the complete model (13.7) in the presence of spring damping. Again a dynamical inverse system will be needed, while

the smoothness requirement on $q_d(t)$ would be even more dramatically reduced in that case.

We finally remark that, for robots with flexible links, the inverse dynamics problem gives also rise to an internal dynamics, independently of the presence of modal damping. When specifying a desired motion for the tip of the flexible arm, the associated internal dynamics will be *unstable* and this critical issue has to be tackled to determine a feasible solution.

13.1.3 Regulation Control

We consider next the problem of controlling the motion of a robot with joint elasticity to a *constant* configuration. In this problem, no trajectory planning is involved and a feedback law should be found that achieves asymptotic stabilization of a desired closed-loop equilibrium. Global solutions (i.e., those valid when starting from any initial state) are indeed preferred.

From the analysis in the previous section, it should be clear that one needs to define only a constant reference q_d (with $\dot{q}_d(t) \equiv 0$) for the link coordinates. From (13.19), a unique reference θ_d for the motor variables is in fact associated with the desired q_d (which, in turn, may result from a desired pose of the robot end-effector). Furthermore, (13.20) provides the needed static torque to be applied at steady state by any feasible controller.

A major aspect of the presence of joint elasticity is that the feedback part of the control law can depend in general on *four* variables for each joint: the motor and link position, and the motor and link velocity. However, in most robots where joint elasticity is not explicitly considered in the system design, at most two sensors are available for measurement at the joints: a position sensor (e.g., an encoder) and, in some cases, a tachometer as a velocity sensor. When no velocity sensors are present, velocity is typically reconstructed by suitable numerical differentiation of high-resolution position measurements. Due to the presence of joint elasticity, the position/velocity quantities that are actually measured depend on where these sensors are mounted in the motor/transmission assembly.

A single link driven through an elastic joint is introduced as a paradigmatic case study to show what can be achieved using different sets of partial state measurements. This simple provides indications on how to handle the general multilink case.

In the absence of gravity, it will be shown that a proportional-derivative (PD) controller based only on motor measurements is sufficient to achieve the desired regulation task. In the presence of gravity, various gravity compensation schemes can be added to the PD feedback controller. These control results mimic the situation of robots with rigid joints, once the convenient quantities for use in the feedback have been identified.

Single Elastic Joint Example

Consider a single link rotating on a horizontal plane (thus, without gravity) and actuated with a motor through an elastic joint coupling (Fig. 13.4). When viscous friction on motor and link side as well as damping across the spring are included, the dynamic model is

$$M\ddot{q} + D(\dot{q} - \dot{\theta}) + K(q - \theta) + F_q\dot{q} = 0,$$

$$B\ddot{\theta} + D(\dot{\theta} - \dot{q}) + K(\theta - q) + F_\theta\dot{\theta} = \tau,$$

where the same, now scalar, notation of Sect. 13.1.1 has been used. Since this system is described by linear equations, Laplace transforms are used to derive the transfer functions of interest, namely from input torque to motor position

$$\frac{\theta(s)}{\tau(s)} = \frac{Ms^2 + (D + F_q)s + K}{\text{den}(s)},$$

and from input torque to link position

$$\frac{q(s)}{\tau(s)} = \frac{Ds + K}{\text{den}(s)},$$

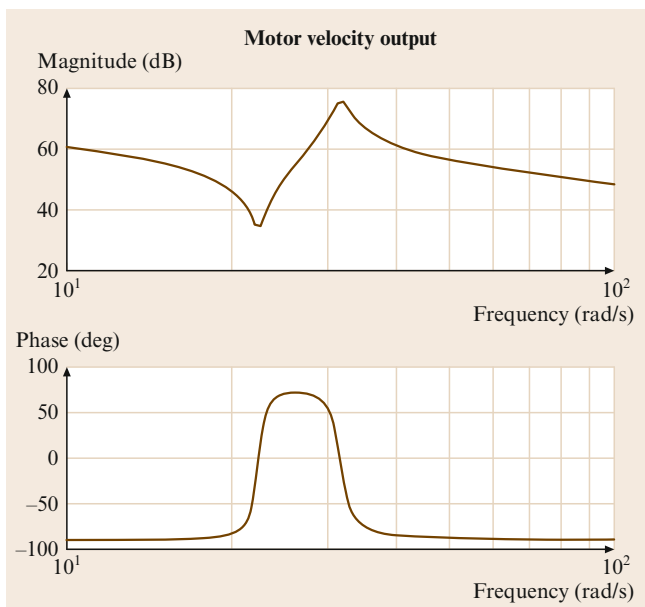


Fig. 13.6 Bode diagrams of the torque to motor velocity transfer function

with the common denominator $\text{den}(s)$ given by

$$\begin{aligned} \text{den}(s) = & \left\{ MB s^3 + [M(D + F_\theta) + B(D + F_q)] s^2 \right. \\ & + [(M + B)K + (F_q + F_\theta)D + F_q F_\theta] s \\ & \left. + (F_q + F_\theta)K \right\} s. \end{aligned}$$

In the case of link position output, the transfer function has a larger relative degree (or pole-zero excess). Figures 13.6 and 13.7 show typical frequency responses in the two cases. For clarity, the outputs have been chosen at the velocity level. In the Bode diagram of the magnitude for the motor velocity output, note the presence of an antiresonance/resonance behavior. Similarly, there is a pure resonance (more pronounced for weak or zero spring damping D) for the link velocity output. The phase profile in Fig. 13.7, with a high-frequency lag of 270° , also indicates the greater control difficulty encountered when closing the feedback loop for link quantities. In experimental tests on a robot joint, such plots are quite characteristic of the presence of concentrated joint elasticity and can be used to assess the relevance of this phenomenon and for identifying model parameters.

For the analysis of the stabilizing properties of different feedback arrangements, all dissipative effects will be neglected ($D = F_q = F_\theta = 0$ is anyway the worst case). Consider first the transfer function relative to the motor position output:

$$\left. \frac{\theta(s)}{\tau(s)} \right|_{\text{no diss}} = \frac{Ms^2 + K}{[MBs^2 + (M + B)K]s^2}. \quad (13.23)$$

As well as a double pole at the origin, this transfer function has a pair of imaginary zeros and poles, with the zero pair occurring at the so-called locked frequency

$$\omega_1 = \sqrt{K/M},$$

which characterizes the oscillations occurring when the motor is locked ($\theta \equiv 0$), e.g., by a high-gain positional feedback. This frequency is used to assess the performance limit of a simple PD control for the motor variables. In order to achieve enough damping in the closed-loop system, the bandwidth should be limited, as a rule of thumb, to one third of ω_1 . Faster transients can be achieved only by taking into account the fourth-order dynamics of the elastic joint assembly. Note also that the frequency of these zeros is always *lower* than that of the poles in (13.23). This is related to the *passivity* of the mapping from torque to motor velocity, a property useful for stability and robust or adaptive control design.

Since the control objective is to regulate the link position output, we are also interested in the open-loop

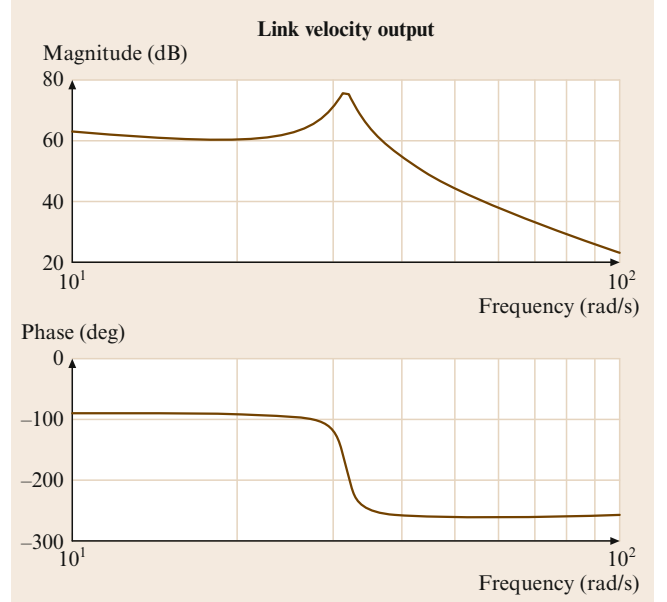


Fig. 13.7 Bode diagrams of the torque to link velocity transfer function

transfer function

$$\left. \frac{q(s)}{\tau(s)} \right|_{\text{no diss}} = \frac{K}{[MBs^2 + (M + B)K]s^2}. \quad (13.24)$$

This transfer function has *no zeros* (in fact, for this it is sufficient to have $D = 0$) so that the relative degree is now four, the maximum possible. We shall see that the nonlinear counterpart of the absence of zeros in (13.24) will play a relevant role also for trajectory control in the multilink case of robots with elastic joints.

It is also worth mentioning that this situation is completely different from the case in which the elastic spring would be moved out of the joint and placed anywhere along the link – a simple one-mode approximate model of link flexibility. In that case the analogous transfer function will possess two symmetric real zeros (a *non-minimum-phase* system), indicating the criticality of a direct inversion of the system input–output map for the execution of a desired link trajectory.

With the desired position given in terms of the link variable q_d , the most natural choice for the design of linear stabilizing feedback using one position and one velocity variable, is to close a PD loop from the link variables,

$$\tau = u_q - (K_{P,q}q + K_{D,q}\dot{q}), \quad (13.25)$$

with position and velocity gains $K_{P,q}$ and $K_{D,q}$, and with $u_q = K_{P,q}q_d$ being the external input used for defining

the set point. It is easy to verify that the closed-loop poles will be unstable no matter how the gains are chosen, so that error feedback from link variables alone should be avoided. In a similar way, also the combination of motor position and link velocity feedback is always unstable.

Another mixed feedback strategy is to use link position and motor velocity:

$$\tau = u_q - (K_{P,q}q + K_{D,m}\dot{\theta}). \quad (13.26)$$

This combination corresponds, e.g., to the case of a tachometer integrated in a direct-current (DC) motor and of an optical encoder placed on the load shaft for sensing its position (without any knowledge about the relevance of joint elasticity). Use of (13.26) leads to the closed-loop characteristic equation

$$BMs^4 + MK_{D,m}s^3 + (B + M)Ks^2 + KK_{D,m}s + KK_{P,q} = 0.$$

Using Routh's criterion, asymptotic stability occurs if and only if the motor velocity gain $K_{D,m} > 0$ and the link position gain satisfies $0 < K_{P,q} < K$, i. e., the proportional feedback should not *override* the spring stiffness. The existence of such an upper bound limits the usefulness of this scheme.

Finally, performing feedback from the motor variables

$$\tau = u_\theta - (K_{P,m}\theta + K_{D,m}\dot{\theta}), \quad (13.27)$$

the closed-loop system will be asymptotically stable provided that both $K_{P,m}$ and $K_{D,m}$ are strictly positive (and otherwise arbitrarily large). This favorable situation lends itself to a convenient generalization in the multilink case.

Note that other partial state feedback combinations would be possible, depending on the available sensing devices. For instance, mounting a strain gauge on the transmission shaft provides a direct measure of the elastic torque $\tau_J = K(\theta - q)$ for control use. Strain gauges are also useful sensors for flexible links. Indeed, full-state feedback may be designed so as to guarantee asymptotic stability and improve the transient behavior considerably; however, this would be obtained at the cost of additional sensors and requires proper tuning of the four gains.

PD Control Using only Motor Variables

For the general multilink case in the absence of gravity, consider the PD control law based on motor position and velocity feedback

$$\tau = K_P(\theta_d - \theta) - K_D\dot{\theta}, \quad (13.28)$$

with symmetric (typically, diagonal) positive-definite gain matrices K_P and K_D . Since $\mathbf{g}(\mathbf{q}) \equiv \mathbf{0}$, it follows from (13.19) that the reference value for the motor position is $\theta_d = \mathbf{q}_d$ (no joint deflection is present nor an input torque is needed at steady state).

The control law (13.28) globally asymptotically stabilizes the desired equilibrium state $\mathbf{q} = \theta = \mathbf{q}_d$, $\dot{\mathbf{q}} = \dot{\theta} = \mathbf{0}$. This can be shown through a Lyapunov argument, completed by the application of La Salle's theorem. In fact, a candidate Lyapunov function is given by the sum of the total energy of the system (kinetic plus elastic potential) and of the control energy due to the proportional term (a virtual elastic potential energy):

$$V = \frac{1}{2} \dot{\theta}^\top \mathcal{M}(\theta) \dot{\theta} + \frac{1}{2} (\mathbf{q} - \theta)^\top \mathbf{K} (\mathbf{q} - \theta) + \frac{1}{2} (\theta_d - \theta)^\top \mathbf{K}_P (\theta_d - \theta) \geq 0. \quad (13.29)$$

Computing the time derivative of V along the trajectories of the closed-loop system given by (13.7) (or (13.9)) and (13.28), and taking into account the skew-symmetry of $\dot{\mathcal{M}} - 2\mathcal{C}$, leads to

$$\dot{V} = -\dot{\theta}^\top \mathbf{K}_D \dot{\theta} \leq 0.$$

The inclusion of dissipative terms (viscous friction and spring damping) would render \dot{V} even more negative-semidefinite. The analysis is completed by verifying that the maximum invariant set contained in the set of states such that $\dot{V} = 0$ (i. e., those with $\dot{\theta} = \mathbf{0}$) collapses into the desired unique equilibrium state.

We point out that an identical control law is able to globally regulate, in the absence of gravity, robots with flexible links to a desired joint configuration. In that case, θ in (13.28) would be the rigid coordinates at the base of the flexible links of the robot.

PD with Constant Gravity Compensation

The presence of gravity requires the addition of some form of gravity compensation to the PD action (13.28). Moreover, this will need an additional structural assumption and may demand some caution in the selection of the control gains.

Before proceeding, we recall a basic property of the gravity vector $\mathbf{g}(\mathbf{q})$ (under assumption A2, the gravity vector in (13.7) is the same as that appearing in the dynamics of the equivalent rigid robot): for robots with rotational joints, elastic or not, a positive constant α exists such that

$$\left\| \frac{\partial \mathbf{g}(\mathbf{q})}{\partial \mathbf{q}} \right\| \leq \alpha, \quad \forall \mathbf{q} \in \mathbb{R}^N. \quad (13.30)$$

The norm of a matrix $A(q)$ is that induced by the Euclidean norm for vectors, i. e., $\|A\| = \sqrt{\lambda_{\max}(A^T A)}$. Inequality (13.30) implies that

$$\|g(q_1) - g(q_2)\| \leq \alpha \|q_1 - q_2\|, \quad \forall q_1, q_2 \in \mathbb{R}^N. \quad (13.31)$$

In common practice, robot joints are never unrealistically *soft*. More precisely, they are stiff enough to have, under the load of the robot's own weight, a unique equilibrium link position q_e associated with any assigned fixed motor position θ_e – the reverse of the relationship expressed by (13.19). This situation is by no means restrictive, but will be stated for clarity as a further modeling assumption:

A5 The lowest joint stiffness is larger than the upper bound on gravity forces acting on the robot, or

$$\min_{i=1, \dots, N} K_i > \alpha.$$

The simplest modification for handling the presence of gravity is to consider the addition of a constant term that compensates exactly the gravity load at the desired steady state. According to (13.19) and (13.20), the control law (13.28) is then modified to

$$\tau = K_P(\theta_d - \theta) - K_D\dot{\theta} + g(q_d), \quad (13.32)$$

with symmetric, typically diagonal, $K_P > 0$ (as a minimum) and $K_D > 0$, and with the motor reference given by $\theta_d = q_d + K^{-1}g(q_d)$.

A sufficient condition guaranteeing that $q = q_d$, $\theta = \theta_d$, $\dot{q} = \dot{\theta} = 0$ will be a globally asymptotically stable equilibrium for the system (13.7) under the control law (13.32) is that

$$\lambda_{\min} \left[\begin{pmatrix} K & -K \\ -K & K + K_P \end{pmatrix} \right] > \alpha, \quad (13.33)$$

with α defined in (13.30). Taking into account the diagonal structure of K and K_P , and thanks to assumption **A5**, it is always possible to fulfill this condition by increasing the smallest proportional gain in the controller (or, the smallest eigenvalue of K_P if this matrix has not been chosen diagonal).

In the following, we sketch the motivation for this condition and the associated proof of asymptotic stability. The equilibrium positions of the closed-loop system are the solutions of

$$\begin{aligned} K(q - \theta) + g(q) &= 0, \\ K(\theta - q) - K_P(\theta_d - \theta) - g(q_d) &= 0. \end{aligned}$$

Indeed, the pair (q_d, θ_d) satisfies these equations. However, in order to obtain a global result one should

guarantee that this pair is the unique solution. Therefore, recalling (13.19), the null term $K(\theta_d - q_d) - g(q_d)$ can be added/subtracted to both equations to obtain

$$\begin{aligned} K(q - q_d) - K(\theta - \theta_d) &= g(q_d) - g(q) \\ -K(q - q_d) + (K + K_P)(\theta - \theta_d) &= 0, \end{aligned}$$

where the matrix in condition (13.33) can be clearly recognized. Taking the norms of terms on both sides and bounding gravity terms using (13.31), the introduced condition (13.33) implies that the pair (q_d, θ_d) is in fact the unique equilibrium. To show asymptotic stability, a candidate Lyapunov function is built based on that used in the absence of gravity (13.29):

$$\begin{aligned} V_{g1} &= V + \mathcal{U}_{\text{grav}}(q) - \mathcal{U}_{\text{grav}}(q_d) - (q - q_d)^T g(q_d) \\ &\quad - \frac{1}{2} g^T(q_d) K^{-1} g(q_d) \geq 0. \end{aligned} \quad (13.34)$$

The last constant term is used to set the minimum value of V_{g1} to zero at the desired equilibrium. The positive-definiteness of V_{g1} and the fact that its unique minimum is at the desired state are again implied by condition (13.33). By the usual computations, it follows that $\dot{V}_{g1} = -\dot{\theta}^T K_D \dot{\theta} \leq 0$ and the result is obtained applying La Salle's theorem.

The control law (13.32) is based only on the knowledge of the gravity term $g(q_d)$ and of the joint stiffness K . The latter appears in fact in the definition of the motor position reference θ_d . Uncertainties in the gravitational term $g(q_d)$ and on the joint stiffness K affect the performance of the controller. Still, the existence of a unique closed-loop equilibrium and its asymptotic stability are preserved when the gravity bound by α is still correct and condition (13.33) holds for the true stiffness values. Indeed, the robot will converge to an equilibrium that is different from the desired one – the better the estimates \hat{K} and $\hat{g}(q_d)$ used, the closer the actual equilibrium will be to the desired one.

PD with Online Gravity Compensation

Similarly to the rigid robot case, a better transient behavior is to be expected if gravity compensation (or, more precisely, its perfect cancellation) is performed at all configurations during motion. However, the gravity vector in (13.7) depends on the link variables q , which are assumed not to be measurable at this stage. It is easy to see that using $g(\theta)$, with the measured motor positions in place of the link positions, leads in general to an incorrect closed-loop equilibrium. Moreover, even if q were available, adding $g(q)$ to a motor PD error feedback has no guarantee of success because this compensation, which appears in the motor equation, does

not instantaneously cancel the gravity load acting on the links.

With this in mind, a PD control with *online* gravity compensation can be introduced as follows. Define the variable

$$\tilde{\theta} = \theta - K^{-1}g(q_d), \quad (13.35)$$

which is a *gravity-biased* modification of the measured motor position θ , and let

$$\tau = K_P(\theta_d - \theta) - K_D\dot{\theta} + g(\tilde{\theta}), \quad (13.36)$$

where $K_P > \mathbf{0}$ and $K_D > \mathbf{0}$ are both symmetric (and typically diagonal) matrices. The control law (13.36) can still be implemented using only motor variables. The term $g(\tilde{\theta})$ only provides an approximate cancellation (though of a large part) of gravity during motion, but leads to the correct gravity compensation in the steady state. In fact, by using (13.19) and (13.35), it follows that

$$\tilde{\theta}_d := \theta_d - K^{-1}g(q_d) = q_d,$$

so that $g(\tilde{\theta}_d) = g(q_d)$.

Global asymptotic stability of the desired equilibrium can be guaranteed under the same condition (13.33) used for the constant gravity compensation case. A slightly different Lyapunov candidate is defined, starting again from the one in (13.29), as

$$V_{g2} = V + \mathcal{U}_{\text{grav}}(q) - \mathcal{U}_{\text{grav}}(\tilde{\theta}) - \frac{1}{2} \mathbf{g}^\top(q_d) K^{-1} \mathbf{g}(q_d) \geq 0,$$

to be compared with (13.34).

The use of an online gravity compensation scheme typically provides a smoother time course and a noticeable reduction of the positional transient errors, with no additional control effort in terms of peak and average torques. We note that choices of low position gains, even when largely violating the sufficient condition (13.33) for stability, may still work, contrary to the case of constant gravity compensation. However also for increasing values of joint stiffness, in the limit for $K \rightarrow \infty$, the range of feasible values of K_P that guarantee exact regulation does not extend down to zero.

A possibility to refine the online gravity compensation scheme, again based only on motor position measurement, is offered by the use of a fast iterative algorithm that elaborates the measure θ in order to generate a quasistatic estimate $\bar{q}(\theta)$ of the current (unmeasured) q . In fact, in any steady-state configuration (q_s, θ_s), a direct mapping is defined from q_s to θ_s :

$$\theta_s = h_g(q_s) := q_s + K^{-1}g(q_s).$$

Assumption A5 is sufficient to guarantee the existence and uniqueness also of the inverse mapping $q_s = h_g^{-1}(\theta_s)$. For a measured θ , the function

$$q = T(q) := \theta - K^{-1}g(q)$$

is then a *contraction mapping* and the iteration

$$q_{i+1} = T(q_i), \quad i = 0, 1, 2, \dots$$

will converge to the unique fixed point of this mapping, which is exactly the value $\bar{q}(\theta) = h_g^{-1}(\theta)$. A suitable initialization for q_0 is either the measured θ or the value \bar{q} computed at the previous sampling instant. In this way, just two or three iterations are needed to obtain sufficient accuracy and this is fast enough to be implemented within a sensing/control sampling interval of a digital robot controller.

With this iterative scheme running in the background, the regulation control law becomes

$$\tau = K_P(\theta_d - \theta) - K_D\dot{\theta} + g[\bar{q}(\theta)], \quad (13.37)$$

with symmetric (diagonal) $K_P > \mathbf{0}$ and $K_D > \mathbf{0}$. A proof of the global asymptotic stability of this control scheme can be given by further modifying the previous Lyapunov candidates. The advantage of (13.37) is that any positive value of the feedback gain K_P is allowed, thus recovering in full the operative working conditions of the rigid case with exact gravity cancellation.

Both online gravity compensation schemes (13.36) and (13.37) realize a *compliance control* in the joint space with only motor measurements. The same idea can also be extended to the case of Cartesian compliance control by evaluating the direct kinematics and the Jacobian (transpose) of the robot arm with $\tilde{\theta}$ or, respectively, $\bar{q}(\theta)$ in place of q .

Full-State Feedback

When the feedback law is based on a full set of measurements of the robot state, the transient performance of regulation control laws can be improved. Taking advantage of the availability of a joint torque sensor, we present a convenient design for the reduced model (13.9) of robots with elastic joints, including spring damping as a dissipation effect. Full-state feedback can be obtained in two phases, by combining a preliminary torque feedback with the motor feedback law in (13.28).

Using $\tau_J = K(\theta - q)$, the motor equation can be rewritten as

$$B\ddot{\theta} + \tau_J + DK^{-1}\dot{q} = \tau.$$

The joint torque feedback

$$\tau = \mathbf{B}\mathbf{B}_\theta^{-1}\mathbf{u} + (\mathbf{I} - \mathbf{B}\mathbf{B}_\theta^{-1})(\tau_J + \mathbf{D}\mathbf{K}^{-1}\dot{\tau}_J), \quad (13.38)$$

where \mathbf{u} is an auxiliary input to be designed, transforms the motor equations into

$$\mathbf{B}_\theta\ddot{\theta} + \tau_J + \mathbf{D}\mathbf{K}^{-1}\dot{\tau}_J = \mathbf{u}.$$

In this way, the apparent motor inertia can be reduced to a desired, arbitrary small value \mathbf{B}_θ , with clear benefits in terms of vibration damping. For instance in the linear and scalar case, a very small B shifts the pair of complex poles in (13.23) to a very high frequency, with the joint behaving almost as a rigid one.

Setting

$$\mathbf{u} = \mathbf{K}_{P,\theta}(\theta_d - \theta) - \mathbf{K}_{D,\theta}\dot{\theta} + \mathbf{g}(q_d)$$

in (13.38) leads to the state feedback controller

$$\begin{aligned} \tau &= \mathbf{K}_P(\theta_d - \theta) - \mathbf{K}_D\dot{\theta} + \mathbf{K}_T[\mathbf{g}(q_d) - \tau_J] \\ &\quad - \mathbf{K}_S\dot{\tau}_J + \mathbf{g}(q_d), \end{aligned} \quad (13.39)$$

with gains

$$\mathbf{K}_P = \mathbf{B}\mathbf{B}_\theta^{-1}\mathbf{K}_{P,\theta},$$

$$\mathbf{K}_D = \mathbf{B}\mathbf{B}_\theta^{-1}\mathbf{K}_{D,\theta},$$

$$\mathbf{K}_T = \mathbf{B}\mathbf{B}_\theta^{-1} - \mathbf{I}$$

and

$$\mathbf{K}_S = (\mathbf{B}\mathbf{B}_\theta^{-1} - \mathbf{I})\mathbf{D}\mathbf{K}^{-1}.$$

Indeed, the law (13.39) can also be rewritten in terms of $(\theta, q, \dot{\theta}, \dot{q})$, if the torque sensor is not available. However, keeping this structure of the gains preserves the interesting physical interpretation of what the full-state feedback controller is achieving.

13.1.4 Trajectory Tracking

As for rigid robot arms, the problem of tracking desired time-varying trajectories is harder for robots with elastic joints than achieving constant regulation. In general, solving this problem requires the use of full-state feedback and knowledge of all the terms in the dynamic model.

Under these conditions, we shall focus on the *feedback linearization* approach, namely a nonlinear state feedback law that leads to a closed-loop system with decoupled and exactly linear behavior for all the N degrees of freedom of the robot (in fact, the link variables q). The tracking errors along the reference trajectory are forced to be *globally exponentially stable*, with a decaying rate that can be directly specified through the choice of the

scalar feedback gains in the controller. This fundamental result is the direct extension of the well-known *computed torque* method for rigid robots. Because of its relevant properties, the feedback linearization approach can be used as a reference to assess the performance of any other trajectory tracking control law, which may possibly be designed using less/approximate model information and/or only partial-state feedback.

However, in the presence of joint elasticity, the design of a feedback linearization law is not straightforward. Furthermore, as soon as $\mathbf{S} \neq \mathbf{0}$, the dynamic model (13.7) will not satisfy the existing necessary conditions for exact linearization (nor those for input–output decoupling) when only a *static* (or, instantaneous) feedback law from the full state is allowed. Therefore, we will restrict our attention to the more tractable case of the reduced dynamic model (13.9) and sketch only briefly the more general picture.

As a second much simpler approach to trajectory tracking problems, we also present a *linear control design* that makes use of a model-based feedforward command and a precomputed state reference trajectory, as obtained from the inverse dynamics computations of Sect. 13.1.2, with the addition of a linear feedback from the full state. In this case, convergence to the desired trajectory is only locally guaranteed, i.e., the tracking errors should be small enough, but the control implementation is straightforward and the real-time computational burden is considerably reduced.

Feedback Linearization

Consider the reduced model (13.9) and let the reference trajectory be specified by a desired smooth link motion $q_d(t)$. The control design will proceed by system inversion, in a similar way to the inverse dynamics computations of Sect. 13.1.2 but using now the current measures of the state variables $(q, \theta, \dot{q}, \dot{\theta})$ instead of the reference state evolution $(q_d, \theta_d, \dot{q}_d, \dot{\theta}_d)$. Notably, there is no need to transform the robot equations into their state-space description, which is the standard form used in control design for general nonlinear systems; we will use the robot model directly in its second-order differential form (typical of mechanical systems).

The outcome of the inversion procedure will be the definition of a torque τ , in the form of a static state feedback control law, which cancels the original robot dynamics and replaces it with a desired linear and decoupled one of suitable differential order. In this sense, the control law *stiffens* the dynamics of the robot with elastic joints. The feasibility of inverting the system from the chosen output q without causing instability prob-

lems (related to the presence of unobservable dynamics in the closed-loop system, after cancellation) is a relevant property of robots with elastic joints. In fact, this is the direct generalization to the nonlinear multiple-input multiple-output (MIMO) case of the possibility of inverting a scalar transfer function in the absence of zeros (see also the considerations made on (13.24)).

Rewrite the link equation in the compact form

$$M(q)\ddot{q} + n(q, \dot{q}) + K(q - \theta) = 0, \quad (13.40)$$

where again we set $n(q, \dot{q}) = c(q, \dot{q}) + g(q)$. None of the above quantities depends instantaneously on the input torque τ . Therefore, we can differentiate once and obtain

$$M(q)q^{[3]} + \dot{M}(q)\ddot{q} + \dot{n}(q, \dot{q}) + K(\dot{q} - \dot{\theta}) = 0. \quad (13.41)$$

Proceeding one step further leads to

$$M(q)q^{[4]} + 2\dot{M}(q)q^{[3]} + \ddot{M}(q)\ddot{q} + \ddot{n}(q, \dot{q}) + K(\ddot{q} - \ddot{\theta}) = 0, \quad (13.42)$$

where $\ddot{\theta}$ now appears. The motor acceleration is at the same differential level of τ in the motor equation

$$B\ddot{\theta} + K(\theta - q) = \tau, \quad (13.43)$$

and thus, replacing $\ddot{\theta}$ from (13.43), we get

$$M(q)q^{[4]} + 2\dot{M}(q)q^{[3]} + \ddot{M}(q)\ddot{q} + \ddot{n}(q, \dot{q}) + K\ddot{q} = KB^{-1}[\tau - K(\theta - q)]. \quad (13.44)$$

We note that, using (13.40), the last term $K(\theta - q)$ in (13.44) may also be replaced by $M(q)\ddot{q} + n(q, \dot{q})$.

Since the matrix $A(q) = M^{-1}(q)KB^{-1}$ is always nonsingular, an arbitrary value v can be assigned to the fourth derivative of q by a suitable choice of the input torque τ . The matrix $A(q)$ is the so-called decoupling matrix of the system and its nonsingularity is a necessary and sufficient condition for imposing a decoupled input-output behavior by nonlinear static state feedback. Moreover, (13.44) indicates that each component q_i of q needs to be differentiated $r_i = 4$ times in order to be algebraically related to the input torque τ (r_i is the relative degree of q_i , when this is chosen as a system output). Since there are N link variables, the total sum of the relative degrees is $4N$, equal to the dimension of the state of a robot with elastic joints. All these facts taken together lead to the conclusion that, when inverting (13.44) to determine the input τ that imposes $q^{[4]} = v$, there will be no other dynamics left other than the one appearing in the closed-loop input-output map.

Therefore, choose

$$\tau = BK^{-1} \left[M(q)v + \alpha(q, \dot{q}, \ddot{q}, q^{[3]}) \right] + [M(q) + B]\ddot{q} + n(q, \dot{q}), \quad (13.45)$$

with

$$\alpha(q, \dot{q}, \ddot{q}, q^{[3]}) = \ddot{M}(q)\ddot{q} + 2\dot{M}(q)q^{[3]} + \ddot{n}(q, \dot{q}),$$

where the terms in α have been ordered according to the dependence on increasing orders of derivatives of q . The control law (13.45) leads to a closed-loop system fully described by

$$q^{[4]} = v, \quad (13.46)$$

i.e., chains of four input-output integrators from each auxiliary input v_i to each link position output q_i , for $i = 1, \dots, N$. The robot system thus has been exactly linearized and decoupled by the nonlinear feedback law (13.45).

The complete control law (13.45) is expressed as a function of the so-called *linearizing* coordinates $(q, \dot{q}, \ddot{q}, q^{[3]})$ only. This has led to some misunderstandings in the past, as it seemed that the feedback linearization approach for robots with elastic joints would need direct measures of the link acceleration \ddot{q} and jerk $q^{[3]}$, which are impossible to obtain with currently available sensors (or would require multiple numerical differentiations of position measures in real time, with critical noise problems).

When considering the latest technology in the field, it is now feasible to have a set of sensors for elastic joints measuring in a reliable and accurate way the motor position θ (and, possibly, also its velocity $\dot{\theta}$), the joint torque $\tau_j = K(\theta - q)$, as well as the link position q . For instance, this is the arrangement of sensors available at each joint of the *LWR-III* lightweight manipulator, where Hall sensors are used for the motor position, joint torque sensing is based on strain gauges, and a high-end capacitive sensor is used for the link position (Fig. 13.8). Therefore, only one numerical differentiation is needed in order to obtain a good estimate of \dot{q} and/or $\dot{\tau}_j$ as well. Note that, depending on the specific sensor resolution, it may also be convenient to evaluate q using the measures of θ and τ_j as $\theta - K^{-1}\tau_j$.

With this in mind, it is easy to see that the following three sets of $4N$ variables

$$(q, \dot{q}, \ddot{q}, q^{[3]}), \quad (q, \theta, \dot{q}, \dot{\theta}), \quad (q, \tau_j, \dot{q}, \dot{\tau}_j)$$

are all equivalent state variables for a robot with elastic joints, related by globally invertible transformations.

Therefore, under the assumption that the dynamic model is available, we can completely rewrite the feedback linearizing control law (13.45) in terms of the more *conventional* state $(q, \theta, \dot{q}, \dot{\theta})$ or, taking advantage of a joint torque sensor, in terms of $(q, \tau_j, \dot{q}, \dot{\tau}_j)$. In particular, as a byproduct of (13.40) and (13.41), we have

$$\begin{aligned}\ddot{q} &= M^{-1}(q) [K(\theta - q) - n(q, \dot{q})] \\ &= M^{-1}(q) [\tau_j - n(q, \dot{q})],\end{aligned}\quad (13.47)$$

and

$$\begin{aligned}q^{[3]} &= M^{-1}(q) [K(\dot{\theta} - \dot{q}) - \dot{M}(q)\ddot{q} - \dot{n}(q, \dot{q})] \\ &= M^{-1}(q) [\dot{\tau}_j - \dot{M}(q)\ddot{q} - \dot{n}(q, \dot{q})],\end{aligned}\quad (13.48)$$

where the acceleration \ddot{q} appearing in (13.48) has already been computed through (13.47). Therefore, the exact linearizing and decoupling control law can be rewritten in terms of a static state feedback law of the form $\tau = \tau(q, \theta, \dot{q}, \dot{\theta}, v)$ or $\tau = \tau(q, \tau_j, \dot{q}, \dot{\tau}_j, v)$. Indeed, the evaluation of the various derivatives of the dynamic model terms that are present in these expressions should be properly organized in order to save computational time.

Based on the resulting (13.46), the trajectory-tracking problem is solved by setting

$$\begin{aligned}v &= q_d^{[4]} + K_3(q_d^{[3]} - q^{[3]}) + K_2(\ddot{q}_d - \ddot{q}) \\ &\quad + K_1(\dot{q}_d - \dot{q}) + K_0(q_d - q),\end{aligned}\quad (13.49)$$

where it is assumed that the reference trajectory $q_d(t)$ is (at least) three times continuously differentiable (i.e., the fourth derivative $q_d^{[4]}$ exists), and the *diagonal* matrices K_0, \dots, K_3 have elements such that

$$s^4 + K_{3,i}s^3 + K_{2,i}s^2 + K_{1,i}s + K_{0,i}, \quad i=1, \dots, N,$$

are Hurwitz polynomials. In view of the obtained decoupling, the trajectory position error $e_i(t) = q_{d,i}(t) - q_i(t)$ for the i -th link then satisfies

$$e_i^{[4]} + K_{3,i}e_i^{[3]} + K_{2,i}\ddot{e}_i + K_{1,i}\dot{e}_i + K_{0,i}e_i = 0;$$

yielding $e_i(t) \rightarrow 0$ in a global exponential way, for any initial state. A series of remarks are in order.

- When the initial state $(q(0), \theta(0), \dot{q}(0), \dot{\theta}(0))$ is matched with the reference trajectory and its first three derivatives at time $t=0$ (for this check, (13.47, 48) are to be used), exact reproduction of the reference trajectory is achieved at all times.
- In the case of a discontinuity of the reference position trajectory, or of any of its first three derivatives, at a time $t^* \in [0, T]$, a trajectory error occurs at $t = t^*$

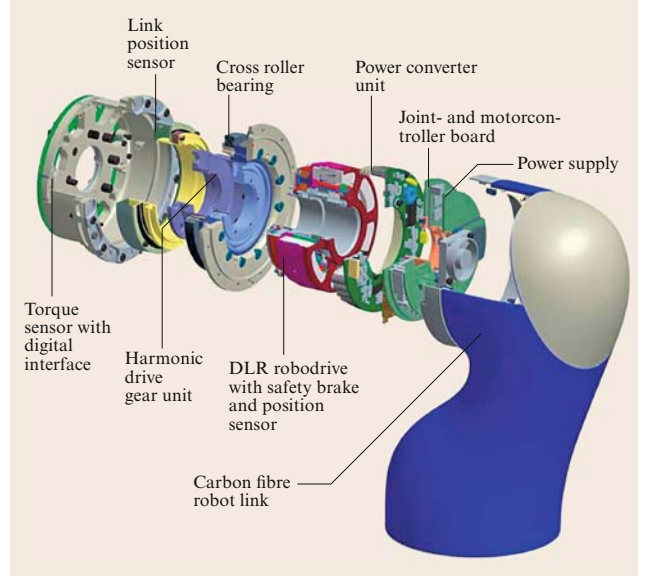


Fig. 13.8 Exploded view of a joint of the *DLR LWR-III* lightweight manipulator and its sensor suite

that will again decay to zero, independently for each link and with the prescribed exponential rate.

- The choice of the gains $K_{3,i}, \dots, K_{0,i}$ can be made by a pole placement procedure (equivalent in this case to an eigenvalue assignment). Let $\lambda_1, \dots, \lambda_4$ be four poles with negative real parts, possibly given in complex pairs and/or coincident, specifying the desired transient of the trajectory error. These closed-loop poles will be assigned by the unique choice of real and positive gains

$$K_{3,i} = -(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4),$$

$$K_{2,i} = \lambda_1(\lambda_2 + \lambda_3 + \lambda_4) + \lambda_2(\lambda_3 + \lambda_4) + \lambda_3\lambda_4,$$

$$K_{1,i} = -[\lambda_1\lambda_2(\lambda_3 + \lambda_4) + \lambda_3\lambda_4(\lambda_1 + \lambda_2)],$$

$$K_{0,i} = \lambda_1\lambda_2\lambda_3\lambda_4.$$

When the values of link and motor inertias are very different from each other, or when the joint stiffness is very large, the above fixed choice of gains has the drawback of generating too large control efforts. In those cases, a more tailored set of eigenvalues can be assigned scheduling their placement as a function of the physical data of robot inertias and joint stiffnesses.

- When compared to the *computed torque* method for rigid robots, the feedback linearization control for trajectory tracking given by (13.45, 47, 48, 49) requires the inversion of the inertia matrix $M(q)$ and

the additional evaluation of derivatives of the inertia matrix and of other terms in the dynamic model.

- The feedback linearization approach can also be applied without any changes in the presence of viscous (or otherwise smooth) friction at the motor and link side. The inclusion of spring damping leads instead to a third-order decoupled differential relation between the auxiliary input v and q , leaving thus a N -dimensional unobservable but asymptotically stable dynamics in the closed-loop system. In this case, only input–output (and not full-state) linearization and decoupling is achieved.

We conclude with some considerations on exact linearization/decoupling for the general model (13.7) of robots with elastic joints. Unfortunately, due to the presence of the inertial coupling matrix $S(q)$ between links and rotors, the above control design can no longer be applied. Consider as an illustration the case of a constant S in (13.7), with the associated model simplifications. Solving for $\ddot{\theta}$ from the motor equations, $\ddot{\theta} = B^{-1}(\tau - \tau_j - S^T \ddot{q})$, and substituting into the link equations leads to

$$\begin{aligned} [M(q) - SB^{-1}S^T] \ddot{q} + n(q, \dot{q}) - (I + SB^{-1}) \tau_j \\ = -SB^{-1} \tau. \end{aligned}$$

In this case, the input torque τ appears already in the expression for the link acceleration \ddot{q} . Using (13.6), the expression of the decoupling matrix is then

$$A(q) = -[M_L(q) + M_R(q)]^{-1} SB^{-1},$$

which is never full rank in view of the structure (13.5) of S . As a consequence, the necessary condition for obtaining (at least) input–output linearization and decoupling by static state feedback fails to hold.

However, by resorting to the use of a larger class of control laws, it is still possible to obtain an exact linearization and decoupling result. For this purpose, consider a *dynamic* state feedback controller of the form

$$\begin{aligned} \tau &= \alpha(q, \dot{q}, \theta, \dot{\theta}, \xi) + \beta(q, \dot{q}, \theta, \dot{\theta}, \xi)v, \\ \dot{\xi} &= \gamma(q, \dot{q}, \theta, \dot{\theta}, \xi) + \delta(q, \dot{q}, \theta, \dot{\theta}, \xi)v, \end{aligned} \quad (13.50)$$

where $\xi \in \mathbb{R}^v$ is the state of the dynamic compensator, α , β , γ , and δ are suitable nonlinear vector functions, and $v \in \mathbb{R}^N$ is (as before) an external input used for trajectory-tracking purposes. It is possible to show in general that a dynamic compensator (13.50) of order at most $v = 2N(N-1)$ can be designed so as to yield

a closed-loop system that is globally described by

$$q^{[2(N+1)]} = v,$$

in place of (13.46). Note the coincidence of this differential order with the one found for exact reproducibility of a desired trajectory $q_d(t)$ in Sect. 13.1.2. The tracking problem is then solved by a direct generalization of the linear stabilization design of (13.49).

It is interesting to give a physical interpretation for this control result. The structural obstruction to input–output decoupling of the model (13.7) is that the motion of a link attached to an elastic joint is affected *too soon* (at the second differential level) by torques originating at other elastic joints. This is due to the inertial couplings present between the motors and links. The addition of dynamics (i.e., of integrators) in the controller slows down these low-energy paths and allows for the high-energy effects of the elastic torque at the local joint (a fourth-order differential path) to come into play. This dynamic balancing allows both input–output decoupling and exact linearization of the extended system (having the robot and the controller states).

Linear Control Design

The feedback linearization approach to trajectory tracking gives rise to a rather complex nonlinear control law. Its main advantage of globally enforcing a linear and decoupled behavior on the trajectory error dynamics can be traded off with a control design that achieves only local stability around the reference trajectory, but is much simpler to implement (and may run at higher sampling rates).

For this, the inverse dynamics results of Sect. 13.1.2 can be used, assuming either the reduced or the complete robot model and with or without dissipative terms. Given a sufficiently smooth desired link trajectory $q_d(t)$, we have seen that it is always possible to associate:

1. the nominal torque $\tau_d(t)$ needed for its exact reproduction, and
2. the reference evolution of all other state variables (e.g., $\theta_d(t)$, as given by (13.13), or $\tau_{j,d}(t)$).

These signals define a sort of steady-state operation for the system.

A simpler tracking controller combines a model-based feedforward term with a linear feedback term using the trajectory error: the linear feedback locally stabilizes the system around the reference state trajectory, whereas the feedforward torque is responsible for maintaining the robot along the desired motion when the error has vanished.

Using full-state feedback, two possible controllers of this kind are

$$\begin{aligned}\tau = & \tau_d + K_{P,\theta}(\theta_d - \theta) + K_{D,\theta}(\dot{\theta}_d - \dot{\theta}) \\ & + K_{P,q}(q_d - q) + K_{D,q}(\dot{q}_d - \dot{q})\end{aligned}\quad (13.51)$$

and

$$\begin{aligned}\tau = & \tau_d + K_{P,\theta}(\theta_d - \theta) + K_{D,\theta}(\dot{\theta}_d - \dot{\theta}) \\ & + K_{P,J}(\tau_{J,d} - \tau_J) + K_{D,J}(\dot{\tau}_{J,d} - \dot{\tau}_J).\end{aligned}\quad (13.52)$$

These trajectory tracking schemes are the most common in the control practice for robots with elastic joints. In the absence of full-state measurements, they can be combined with an observer of the unmeasurable quantities. An even simpler realization is

$$\tau = \tau_d + K_P(\theta_d - \theta) + K_D(\dot{\theta}_d - \dot{\theta}), \quad (13.53)$$

which uses only motor measurements and relies on the results obtained for the regulation case.

The different gain matrices used in (13.51, 52, 53) have to be tuned using a linear approximation of the robot system. This approximation may be obtained at a fixed equilibrium point or around the actual reference trajectory, leading respectively to a linear time-invariant or to a linear time-varying system. While the existence of (possibly time-varying) stabilizing feedback matrices is guaranteed by the controllability of these linear approximations, the validity of the approach is only *local* in nature and the region of convergence will depend both on the given trajectory and on the robustness of the designed linear feedback.

It should be mentioned that such a control approach to trajectory tracking problems can also be used for robots with flexible links. Once the inverse dynamics problem has been solved (for the case of link flexibility, this typically results in a noncausal solution), a controller of the form (13.53) (or (13.51), with link deflection δ and deflection rate $\dot{\delta}$ replacing, respectively, q and \dot{q}), can be directly applied.

13.1.5 Further Reading

This section contains the main references for the part of this chapter on robots with joint flexibility. In addition, we indicate to a larger bibliography for topics that have not been included here.

Early interest in problems arising from the presence of flexible transmissions in robot manipulators dates back to [13.1, 2], with first experimental findings on the GE P-50 arm. Relevant mechanical considerations involved in the design of robot arms and in the evaluation of their compliant elements can be found in [13.3].

One of the first studies on the inclusion of joint elasticity in the dynamic modeling of robot arms is due to [13.4]. The detailed analysis of the model structure presented in Sect. 13.1.1 comes from [13.5], with some updates from [13.6]. The simplifying assumption leading to the reduced model was introduced in [13.7]. The special case of motors mounted on the driven links is considered in [13.8]. The observation that joints with limited elasticity lead to a singularly perturbed dynamic model was first recognized in [13.9]. Symbolic manipulation programs for the automatic generation of dynamic models of robots with elastic joints were also developed very early [13.10].

The inverse dynamics computations in Sect. 13.1.2 can be traced back to [13.11]. The use of programs such as Modelica for the numerical evaluation of inverse dynamics is highlighted in [13.6].

The first specific control designs were based on decentralized linear controllers, see, e.g., [13.12] and [13.13], where a fourth-order dynamics local to each elastic joint was used. However, schemes with proved global convergence characteristics appeared only later.

In Sect. 13.1.3, the PD controller with constant gravity compensation is a contribution of [13.14]. This was also extended to the case of robots with flexible links in [13.15]. The two versions of regulation control with online gravity compensation are due, respectively, to [13.16] and [13.17]. Both these control laws were extended to Cartesian compliance schemes in [13.18] and [13.19]. A general regulation framework based on energy shaping has been proposed in [13.20]. In the absence of any information on gravity, a PID regulator with semiglobal stability properties has been presented in [13.21], while a global learning scheme was proposed in [13.22].

The presentation of a full-state feedback design for regulation (and tracking) follows the idea used in [13.23]. Special interest has been devoted over the years to joint torque feedback, from the placement of torque sensors on transmission shafts [13.24] or within harmonic drives [13.25] to their use for achieving robust control performance [13.26].

The fact that the reduced model of robots with elastic joints can always be feedback linearized via static state feedback was shown for the first time in [13.7]. A previous similar result for a specific robot kinematics had been found in [13.27]. The discrete-time implementation of the feedback linearization approach has been studied in [13.28], while its robustness properties were analyzed in [13.29]. Feedback linearization and input-output decoupling were also considered for the case of

viscoelastic joints in [13.30], and for robots with joints of mixed type, some rigid and some elastic, in [13.31]. The same idea of inversion control, leading to input–output decoupling and linearization, has also been successfully applied to the tracking problem of joint-level trajectories in robots with flexible links [13.32].

For the general model of robots with elastic joints, the use of dynamic state feedback was first proposed in [13.33]. A comparative study of the errors induced by neglecting the motor–link inertia couplings (thus, for robots that are not linearizable by static state feedback) was carried out in [13.34]. The general algorithm for constructing the dynamic linearizing feedback is presented in [13.35].

Other control design methodologies available for trajectory tracking, which were not discussed in Sect. 13.1.4, are based on singular perturbation, backstepping, or passivity. Nonlinear controllers based on the two-time scale separation property were proposed by [13.36] and [13.37]. These corrective controllers are an outcome of the singular perturbation model form and should be preferred in the case of very stiff joints, as they do not generate high-gain laws in the limit. Backstepping is based on the idea of considering the joint elastic torque [13.38] or the motor position [13.39] as an intermediate fictitious input to

be used to control the link equations, and then to design the actual torque input in the motor equations so as to follow the reference behavior for the previous intermediate input. The main advantage is that this design may be easier to be transformed into an adaptive version. Adaptive control results for robots with elastic joints include the high-gain (approximate) scheme [13.40] and the global (but very complex) solution obtained in [13.41], both analyzed using only the reduced dynamic model. Moreover, robust control schemes have been proposed in [13.42], based on sliding mode techniques, and in [13.43], using iterative learning on repetitive tasks.

Different state observers have been presented, starting from an approximate one in [13.44] up to the exact ones in [13.45] and [13.5], in which a tracking controller based on the estimated state was also tested. In all cases, link position or link position and velocity measurements are assumed. More recently, the use of motor position and link acceleration has been shown to be a feasible alternative for robust state reconstruction [13.46].

Finally, the force control problem in constrained tasks for robots with elastic joints has been tackled following a singular perturbation technique [13.47, 48], an inverse dynamics approach [13.49], or an adaptive strategy [13.50].

13.2 Robots with Flexible Links

Arm flexibility is a dynamic behavior in which kinetic energy interacts with elastic potential energy. Kinetic energy is stored in moving inertia, and potential energy is stored in compliant members. Flexible links provide both the inertia and compliance distributed throughout the member. Flexibility manifests itself as mechanical oscillations and static deflections, greatly complicating the motion control of a mechanical arm. If the time to settle the oscillations is significant relative to the cycle time of the overall task, flexibility will be a major consideration in the arm design.

13.2.1 Design Issues

We will deal with these issues in a modular way, separating link flexibility and joint flexibility. The first topic of link flexibility deals with the inherent spatial distribution of compliance and inertia, perhaps with both effects existing in the same element of material. The links of a manipulator arm can usually be stiffened by adding structural mass or by improving the material properties

or distribution over the links. At some points the links may be treated as rigid. The designer must realize that an arm with heavy rigid links and flexible joints may be more flexible dynamically than a lighter arm with more link flexibility and less inertia. The consequences are predicted by the natural frequency of the first flexible mode of motion, with the joint actuators locked, which can be approximated by

$$\omega_1 = \sqrt{\frac{k_{\text{eff}}}{I_{\text{eff}}}} \quad (13.54)$$

where k_{eff} is the effective spring constant and I_{eff} is the effective inertia. For the elastic joint case k_{eff} and I_{eff} are directly available from the elastic joint parameters and mass properties of the arm. The *effective* values applicable to a simple beam in bending, for example, would result from boundary conditions that are clamped at one end and free at the other and would be selected to yield a good approximation to the natural frequency. This natural frequency is readily available from handbooks on

vibration in this case as $\omega_1 = 3.52\sqrt{EI/ml^4}$. (The variables are used in (13.55) and (13.56) and defined below.) More-complex geometries can be evaluated by various means, including the transfer matrix method discussed in Sect. 13.2.2 on modeling. The arm should be considered flexible when this frequency is low enough to interfere with the design of controllers based on rigid assumptions. The common proportional-derivative (PD) control of a single joint on a flexible link can only achieve adequate damping of a closed-loop design when the magnitude of the closed loop poles is less than about $\frac{1}{3}$ of ω_1 [13.51].

A distillation of the key aspects of the mechanics of materials is now in order. In some cases static compliance effects represent a member sufficiently, whereas in other cases the true distributed nature of compliance and mass and its dynamics must be modeled. A long structural member with forces f and moments M perpendicular to its long axis is subject to bending. Static bending relates the moment at the axial location x to the displacement w

$$M(x) = EI(x) \frac{\partial^2 w}{\partial x^2}, \quad (13.55)$$

where E is the elastic modulus of the material, and $I(x)$ is the area moment of inertia about the neutral axis of the cross section. The deflection at any point is obtained by integrating this equation from a reference point to a desired point, such as the end of the link.

This is a description of compliance that can be used when mass is isolated from elasticity. If mass is distributed throughout the beam with material mass density per unit volume ρ , the time t must be incorporated so that

$$m(x) \frac{\partial^2 w(x, t)}{\partial t^2} + \frac{\partial^2}{\partial x^2} \left(EI(x) \frac{\partial^2 w(x, t)}{\partial x^2} \right) = 0. \quad (13.56)$$

Here $m(x)$ is the mass density per unit length, incorporating the material properties and the cross-sectional area at x . Assumptions implicit in this equation, which is called the Bernoulli–Euler equation, include the minimal impact of shear distortion and rotational inertia of the cross section of the beam, which is valid for long beams. The Timoshenko beam model relaxes these assumptions but is not included here.

Torsional deflection of an angle θ results if a twisting moment T about the long beam axis occurs, such that, in the static case

$$\theta = \frac{Tl}{JG} = \alpha_{\theta T} T, \quad (13.57)$$

where G is the shear modulus of the material, and J is the polar area moment of inertia about the neutral axis of the beam. Again, the addition of a distributed mass and considering the dynamics produces a partial differential equation with independent variables in space and time:

$$\mu \frac{\partial^2 \theta}{\partial t^2} = GJ \frac{\partial^2 \theta}{\partial x^2}, \quad (13.58)$$

where $\mu(x)$ is the rotary mass moment of inertia per unit length of the shaft. While (13.58) is potentially appropriate for links, the smaller rotational mass moment of inertia of a long slender shaft relative to the links it drives results in the static compliance effects of torsion given by (13.57) being of most importance.

Tension and compression effects should also be acknowledged, although they tend to be the least significant. Here, with deflection in the x -direction, the long axis of the member results in a compliance of

$$\delta = \frac{F_a L}{AE} = \alpha_{XF} F_a \quad (13.59)$$

and a dynamic flexibility of

$$\rho \frac{\partial^2 \xi}{\partial t^2} = E \frac{\partial^2 \xi}{\partial x^2}. \quad (13.60)$$

Note that these effects are special cases of the more general elastic behavior of a member subject to acceleration and external loading, specialized to the member with one long axis. The same general phenomena apply to joint structures (e.g., bearings and couplings) and drive-train components (e.g., gears and cables). The functional constraints on the structural design of these members may be more severe and limit the designer's ability to make them more rigid by adding more material. Under this constraint the combined compliance of the fixed joint design and the rigidized link design results in inferior performance of the combination. A less rigid link with less inertia will raise the lowest natural frequency. Thus the design of the link and the joint must be coordinated.

Flexibility is only one of several link design constraints. Static compliance is obviously another based on the above discussion. Buckling and strength are two more considerations that constrain some of the same design parameters. While it has been shown that flexibility is a dominant constraint for typical arm design regimes [13.52], a simple optimization of bending stiffness of a beam with a tubular geometry illustrates the fallacy of ignoring buckling. If the radius of the tube is varied to maximize bending stiffness with constrained total mass, the radius increases without bound, producing a thin shell of infinitesimal thickness and infinite

radius that lacks robustness. The true constraint becomes local buckling of the tube under slight perturbations of the applied load. A thin metal can illustrates this behavior as it crushes. Strength is another applicable constraint. Various materials differ in their strength, typically represented by their elastic limit or endurance limit on the stress at a point in the member. A sufficiently stiff member may fail due to stress, particularly at points of stress concentration.

In this presentation of options to overcome flexibility, and link flexibility in particular, radically different operational and design strategies should be recognized even though they are not the core of the treatment. The elastic modulus relates stress to strain, but physically the strain rate is sometimes a relevant term, providing structural damping to augment the joint or control-based damping. Composite materials have substantially more damping inherent in their behavior. Damping can be enhanced with passive damping treatments, with constrained layer damping having particularly pronounced effects [13.53]. Smart materials can also be configured to enhance damping [13.54]. An operational strategy known as *bracing* can be combined with redundant actuation to achieve a large workspace for a flexible arm that provides gross motion and a smaller precision workspace after docking the large arm with a stationary structure, as described in [13.55].

The implication above is that flexibility was always due to the moving parts of the arm. However, the arm base itself may be a source of significant flexibility. If the base is stationary, the analysis may be almost identical to that of the flexible links of an arm. However, if the arm is mounted on a vehicle, tires or tracks may be the source of flexibility and empirical evaluation of these new elements might be the best solution. The vehicle could be a boat, aircraft or space vehicle. Here there is no elastic connection to an inertial reference frame and some fundamental changes in approach will be needed. These changes are not unexplored and in some cases are simple extensions of the following approaches.

13.2.2 Modeling of Flexible Link Arms

Mathematical models of link flexibility trade accuracy for ease of use. It is therefore key to determine the effects that are important to represent accurately. The techniques presented here will assume linear elasticity with light damping. Rotational motions must be of modest angular rate so that centrifugal stiffening can be ignored. Small deflections will generally be assumed. These are reasonable assumptions for most robotic devices, but can

be violated in more exotic applications, requiring reevaluation of the assumptions and more tortuous modeling approaches.

Four model types will be examined:

1. lumped-element models having compliance or inertia but not both in a given member
2. finite-element models, which will be only briefly addressed
3. assumed mode models which readily include nonlinear dynamic behavior
4. transfer matrix models that incorporate the true distributed intermingling of compliance and inertia and consequently the infinite-dimensional nature of flexible link arms

With the limited space available here, the discussion will be preliminary but should enable the reader to pursue one or more alternatives knowledgeably.

For lumped elements one may build on the rigid kinematics presented in the section on fundamentals of kinematics (Chap. 1) that establishes rigid transformation matrices A_i . The 4×4 homogenous transformations that describe position can be used to describe deflection as well. Assuming small motions and static behavior (or negligible member mass), the transformation produced by elastic bending, torsion, and compression is

$$E_i = \begin{pmatrix} 1 & -\alpha_{\theta Fi} F_{Yi}^i - \alpha_{\theta Mi} M_{Zi}^i & & \\ \alpha_{\theta Fi} F_{Yi}^i + \alpha_{\theta Mi} M_{Zi}^i & 1 & & \\ \alpha_{\theta Fi} F_{Zi}^i - \alpha_{\theta Mi} M_{Yi}^i & \alpha_{Ti} M_{Xi}^i & & \\ 0 & 0 & & \\ -\alpha_{\theta Fi} F_{Zi}^i + \alpha_{\theta Mi} M_{Yi}^i & \alpha_{Ci} F_{Xi}^i & & \\ -\alpha_{Ti} M_{Xii} & \alpha_{XFi} F_{Yi}^i + \alpha_{XMi} M_{Zi}^i & & \\ 1 & \alpha_{XFi} F_{Zi}^i + \alpha_{XMi} M_{Yi}^i & & \\ 0 & 1 & & \end{pmatrix}, \quad (13.61)$$

where

α_{Ci} = coefficient in compression,
displacement/force,

α_{Ti} = coefficient in torsion, angle/moment,

$\alpha_{\theta Fi}$ = coefficient in bending, angle/force,

$\alpha_{\theta Mi}$ = coefficient in bending, angle/moment,

α_{XFi} = coefficient in bending, displacement/force,

α_{XMi} = coefficient in bending,
displacement/movement,

F_{Xi}^j = the force at the end of link i in the X direction of coordinate frame j ,

F_{Yi}^j = the force at the end of link i in the Y direction of coordinate frame j ,

F_{Zi}^j = the force at the end of link i in the Z direction of coordinate frame j ,

M_{Xi}^j = the moment at the end of link i in the X direction of coordinate frame j ,

M_{Yi}^j = the moment at the end of link i in the Y direction of coordinate frame j ,

M_{Zi}^j = the moment at the end of link i in the Z direction of coordinate frame j .

The listed coefficients depend on the construction of the element and are readily found for a slender beam from simple strength of materials. In other cases a finite-element model or empirical measurements may be more readily used.

Alternating transformations for the undeformed and deformed aspects of N members (links or joints) produces the position vector of the end of arm (EOA):

$$\mathbf{p}^0 = (A_1 E_1 A_2 E_2 \cdots A_i E_i A_{i+1} E_{i+1} \cdots A_N E_N) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \quad (13.62)$$

If the massless elements connect rigid lumped masses a linear spatial model is readily obtained, as described in [13.56]. Servo-controlled joints can also be inserted into this model. This analysis proceeds by examining the deflection of the elastic members produced by joint motion and the resulting forces and moments on the rigid inertias at the ends of the chain of links. The reference concentrates on the special case of only two inertias at each end of the chain but the technique is easily extended to inertias in the middle of the chain as well. The results are six linear second-order equations for each inertia and one first-order equation for each joint.

If the compliance and inertia properties are both to be treated as distributed throughout the same element, the partial differential equations (PDE) introduced in the previous section for bending, torsion, and compression must be employed. Still insisting on a linear treatment, the general solution of these equations can be found in terms of each element with adjoining elements, i. e. the boundary conditions on the PDE. The convenient approach of Laplace transformation of these linear equa-

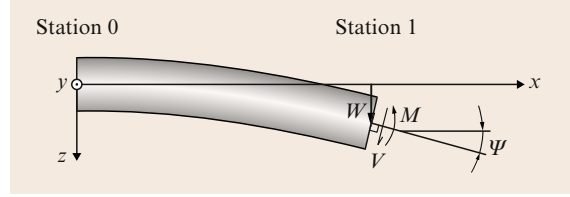


Fig. 13.9 State vector for the bending transfer matrix

tions into the frequency domain, followed by factoring out the boundary conditions into a matrix–vector product, results in a technique known as the transfer matrix method (TMM) [13.57]. This method has been applied productively to general elastomechanics problems as well as specifically to flexible arms [13.58].

The transfer matrix relates variables at two stations along the arm with the number of variables depending on the model complexity. For the planar bending deflection of the beam shown in Fig. 13.9

$$\mathbf{z}_1 = \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_1 = \begin{pmatrix} \text{displacement} \\ \text{angle} \\ \text{moment} \\ \text{shear force} \end{pmatrix}_{\text{at station 1}}, \quad \mathbf{z}_0 = \mathbf{T} \mathbf{z}_1 \quad (13.63)$$

where T is the appropriate element transfer matrix.

If the element is a simple rotary spring with constant k and a damper with constant b then the angle of rotation θ is related to the moment by a differential equation with the following Laplace-domain representation:

$$\begin{aligned} M &= -L(k\theta + b\dot{\theta}) = k\Theta(s) + bs\Theta(s) = M_0 = M_1, \\ W_0 &= W_1, \\ V_0 &= V_1, \\ \Psi_0 &= \Psi_1 - \Theta(s) = \Psi_1 - \frac{1}{k + bs} M_1. \end{aligned}$$

Note that the only variable changed across the element is the angle related to the moment. The transfer matrix representation conventionally uses zero initial conditions and converts to the Fourier representation with $s = i\omega$. This leads to the transfer function

$$\mathbf{T} = \mathbf{C}(i\omega) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \frac{1}{k + b(i\omega)} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

which has the same form of the joint controller shown in equation (13.64).

The beam model is much more complex, and the Euler–Bernoulli model gives

$$B = \begin{pmatrix} c_0 & lc_1 & ac_2 & alc_3 \\ \beta^4 c_3/l & c_0 & ac_1/l & ac_2 \\ \beta^4 c_2/a & \beta^4 lc_3/a & c_0 & lc_1 \\ \beta^4 c_1/al & \beta^4 c_2/a & \beta^4 c_3/l & c_0 \end{pmatrix}, \quad (13.64)$$

where

$$\begin{aligned} \beta^4 &= \omega^2 l^4 \mu / (EI); \quad a = l^2 / (EI), \\ c_0 &= (\cosh \beta + \cos \beta) / 2, \\ c_1 &= (\sinh \beta + \sin \beta) / (2\beta), \\ c_2 &= (\cosh \beta - \cos \beta) / (2\beta^2), \\ c_3 &= (\sinh \beta - \sin \beta) / (2\beta^3), \\ \mu &= \text{density/unit length}, \\ \omega &= \text{circular frequency of vibration}, \\ E &= \text{elastic modulus}, \\ I &= \text{cross-sectional area moment of inertia}. \end{aligned}$$

The spatial variable has been transformed to the spatial Laplace variable but no longer appears explicitly. The Laplace time variable remains and is designated by s or by the frequency variable $\omega = -is$, where $i = \sqrt{-1}$. The other transfer matrices needed for a simple planar model are as follows.

For a rotation in the plane of angle ϕ

$$A = \begin{pmatrix} 1/\cos \phi & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ m_s \omega^2 \sin \phi \tan \phi & 0 & 0 & \cos \phi \end{pmatrix}, \quad (13.65)$$

where m_s is the sum of all outboard masses from the angle to the end of the arm. This is an approximation resulting from ignoring compression elasticity. These

elements appear as an additional mass to the extent they are translated with beam compression.

For a rigid mass, simply applying Newton's laws and collecting term yields

$$R = \begin{pmatrix} 1 & l & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -m\omega^2 l/2 & -I\omega^2 + m\omega^2 l/2 & 1 & l \\ m\omega^2 & -m\omega^2 l/2 & 0 & 1 \end{pmatrix}, \quad (13.66)$$

where m is the mass of the body, I is the mass moment of inertia about an axis through the center of mass and perpendicular to the plane of the arm, l is the length of the mass (the distance between the points of attachment at stations i and $i+1$), and $l/2$ is the distance to the center of mass from station i .

For a controlled joint with the controller transfer function (joint torque/joint angle) = $k(i\omega)$

$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1/k(i\omega) & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (13.67)$$

When combined to represent the planar arm with two joints shown in Fig. 13.10 this analysis yields a composite transfer matrix that relates the four state variables at the two ends of the arm. Further analysis results from imposing the known boundary conditions at these ends.

The transfer matrix representation of an arm presented in [13.59] and pictured in Fig. 13.10 is shown in (13.68).

$$\begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_0 = R_1 B_2 R_3 A_4 C_5 B_6 R_7 \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_7. \quad (13.68)$$

In the example the left end is pinned and the right end is free, yielding

$$\begin{aligned} z_0 = \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_0 &= \begin{pmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ u_{21} & u_{22} & u_{23} & u_{24} \\ u_{31} & u_{32} & u_{33} & u_{34} \\ u_{41} & u_{42} & u_{43} & u_{44} \end{pmatrix} \begin{pmatrix} -W \\ \Psi \\ M \\ V \end{pmatrix}_7, \\ \begin{pmatrix} u_{11} & u_{12} \\ u_{31} & u_{32} \end{pmatrix} \begin{pmatrix} -W \\ \Psi \end{pmatrix}_7 &= \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \end{aligned} \quad (13.69)$$

The determinant of the 2×2 matrix must be zero when ω is a natural frequency, or a system eigenvalue if it is

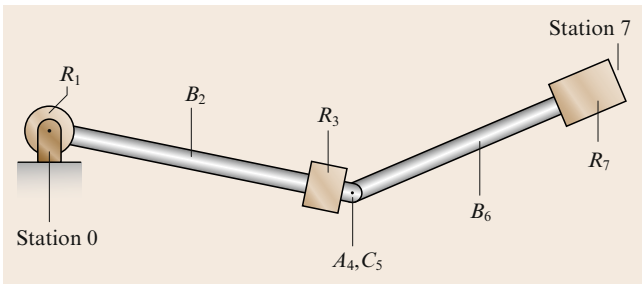


Fig. 13.10 Transfer matrix representation of an arm

complex. This can be found by a numerical search. Evaluating variables (position, angle, shear, and moment) along the length of the arm with the eigenvalues used for the values of ω yields the eigenfunctions. If boundary conditions are not zero but known for given frequencies, the frequency response of the system can be obtained. Hence Bode plots are readily obtained for boundary condition forcing. Interior forcing can be handled using an extended state vector as described in [13.57], which was applied to arms in [13.51] and most recently updated with modern programming techniques in [13.58]. Inverse fast Fourier transform (FFT) will produce a time-domain response from the frequency response.

The creation of a state-space model in the time domain is attractive because it enables the use of powerful state-space design techniques and is compatible with the nonlinear behavior of arms undergoing large motion and experiencing centrifugal and Coriolis forces. The assumed modes method allows such a model to be constructed efficiently. The presentation here follows the early introduction of a recursive manner to calculate these equations introduced in [13.60]. The flexible kinematics of the links must be expressed as a sum of basis functions also known as assumed modes (shapes) $\phi_i(x)$ with time-variable amplitudes $\delta_i(t)$.

$$w(x, t) = \sum_{i=1}^{\infty} \delta_i(t) \phi_i(x). \quad (13.70)$$

The amplitudes and their derivatives become the states of the model. Compatible joint angle variables and their derivatives are also included as the rigid state variables. Joint angles tangent to the ends of the flexible links are compatible with mode shapes based on the clamped boundary conditions at one end and free boundary conditions at the other. This choice is generally chosen for simulation and the rigid coordinates are directly measured by standard joint angle transducers. If rigid coordinates are angles measured between lines connecting successive joint axes, the end of arm is known in terms of rigid coordinates alone and pinned–pinned boundary conditions are consistent. This produces advantages for inverse dynamics calculations as seen shortly.

The flexible and rigid kinematics combined describes the position and velocity of every point on the arm and can be used to express the kinetic energy \mathcal{T} and the potential energy \mathcal{V} . These expressions are used in the conservative form of Lagrange's equation:

$$\frac{d}{dt} \frac{\partial \mathcal{T}}{\partial \dot{q}_i} - \frac{\partial \mathcal{T}}{\partial q_i} + \frac{\partial \mathcal{V}}{\partial q_i} = F_i, \quad (13.71)$$

where F_i is the force that does work as q_i is varied.

The kinematics of the deflection are again described by 4×4 transformation matrices, but this time the assumed mode shapes j are summed to give the position of link i .

$$\mathbf{h}_i^i(\eta) = \begin{pmatrix} \eta \\ 0 \\ 0 \\ 1 \end{pmatrix} + \sum_{j=i}^{m_i} \delta_{ij} \begin{pmatrix} x_{ij}(\eta) \\ y_{ij}(\eta) \\ z_{ij}(\eta) \\ 0 \end{pmatrix}, \quad (13.72)$$

where x_{ij} , y_{ij} , and z_{ij} are the x_i , y_i , and z_i displacement components of mode j of link i 's deflection, respectively, δ_{ij} is the time-varying amplitude of mode j of link i , and m_i is the number of modes used to describe the deflection of link i .

$$\mathbf{E}_i = \left(\mathbf{H}_i + \sum_{j=i}^{m_i} \delta_{ij} \mathbf{M}_{ij} \right), \quad (13.73)$$

where

$$\mathbf{H}_i = \begin{pmatrix} 1 & 0 & 0 & l_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (13.74)$$

and

$$\mathbf{M}_{ij} = \begin{pmatrix} 0 & -\theta_{zij} & \theta_{yij} & x_{ij} \\ \theta_{zij} & 0 & -\theta_{xij} & y_{ij} \\ -\theta_{yij} & \theta_{xij} & 0 & z_{ij} \\ 0 & 0 & 0 & 0 \end{pmatrix}. \quad (13.75)$$

This description is used in the formation of the potential and kinetic energy contributions of each element of mass. The integral over the link accumulates the total energy for that link and the sum over all links gives the energy of the arm system. Interchanging the order of integration and summation allows key invariant parameters to be identified, such as the *modal mass* that multiplies the second derivative of modal amplitude and the *modal stiffness* that multiplies the amplitude itself. The intermediate steps are aimed at the evaluation of Lagrange's equations, which separate into derivatives related to the rigid variables and the flexible variables. If the rigid variables θ are the joint variables, two groups of equations result:

$$1. \quad \frac{d}{dt} \left(\frac{\partial \mathcal{K}}{\partial \dot{\theta}_j} \right) - \frac{\partial \mathcal{K}}{\partial \theta_j} + \frac{\partial \mathcal{V}_e}{\partial \theta_j} + \frac{\partial \mathcal{V}_g}{\partial \theta_j} = F_j \quad (13.76)$$

$$2. \quad \frac{d}{dt} \left(\frac{\partial \mathcal{K}}{\partial \dot{\delta}_{jf}} \right) - \frac{\partial \mathcal{K}}{\partial \delta_{jf}} + \frac{\partial \mathcal{V}_e}{\partial \delta_{jf}} + \frac{\partial \mathcal{V}_g}{\partial \delta_{jf}} = 0, \quad (13.77)$$

where the subscript 'e' indicates elasticity, 'g' indicates gravity, and δ indicates the amplitude of a mode shape.

The simulation form of the equations is achieved by grouping all second derivatives of the rigid and flexible coordinates, which yields

$$\begin{aligned} & \begin{pmatrix} M_{rr}(q) & M_{rf}(q) \\ M_{fr}(q) & M_{ff}(q) \end{pmatrix} \begin{pmatrix} \ddot{\theta} \\ \ddot{\delta} \end{pmatrix} \\ &= - \begin{pmatrix} 0 & 0 \\ 0 & K_s \end{pmatrix} \begin{pmatrix} \theta \\ \delta \end{pmatrix} + N(q, \dot{q}) + G(q) \\ &= R(q, \dot{q}, Q), \end{aligned} \quad (13.78)$$

where θ is a vector of rigid coordinates, usually the joint variables, δ are the flexible coordinates, $q = [\theta^T \delta^T]^T$, M_{ij} is the mass matrix for rigid and flexible coordinates corresponding to the rigid ($i, j = r$) or flexible ($i, j = f$) coordinates and equations, $N(q, \dot{q})$ contains the nonlinear Coriolis and centrifugal terms, $G(q)$ captures gravity effects, Q represents the externally applied forces, and R captures the effect of external forces and all other nonconservative forces including friction.

On the other hand, the inverse dynamics form of the equations is

$$\begin{aligned} Q &= \begin{pmatrix} M_{rr}(q) & M_{rf}(q) \\ M_{fr}(q) & M_{ff}(q) \end{pmatrix} \begin{pmatrix} \ddot{\theta} \\ \ddot{\delta} \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & K_s \end{pmatrix} \begin{pmatrix} \theta \\ \delta \end{pmatrix} \\ &\quad - N(q, \dot{q}) - G(q). \end{aligned} \quad (13.79)$$

In general Q is dependent on the vector T of motor torques or forces at the joints, and the distribution of these effects is such that

$$Q = \begin{pmatrix} B_r \\ B_f \end{pmatrix} T, \quad (13.80)$$

where T has the same dimensionality as q_r . By rearranging the inverse dynamics equation we obtain

$$\begin{pmatrix} B_r & -M_{rf} \\ B_f & -M_{ff} \end{pmatrix} \begin{pmatrix} T \\ \ddot{\delta} \end{pmatrix} = \begin{pmatrix} M_{rr} \\ M_{rf} \end{pmatrix} \ddot{\theta} + N(q, \dot{q}) + G(q) \quad (13.81)$$

which is arranged with a vector of unknowns on the left-hand side of the equation and the prespecified rigid and solvable flexible coordinates on the right. This is at least approximately true, with the caveat that the nonlinear and gravity terms have a weak influence on the immediate flexible coordinates, which are not simultaneously known but which can be estimated for the next time step.

One consequence of the distributed flexibility is non-minimum-phase behavior. This can occur for other cases

where the excitation is separated from the output point by mass and flexibility, a condition known as noncollocation. These systems are characterized by a step response in which the initial movement is opposite in sign from the final movement. This is simply characterized for linear systems as a transfer function zero in the right half-plane.

13.2.3 Control

Sensors for Flexibility Control

Appropriate reaction to flexible dynamics requires appropriate sensors. While a Kalman filter can theoretically observe flexible link states from joint measurements, it is based on back driving the joint from the various modes of flexible dynamics. More-reliable indications of flexibility include strain gauges, accelerometers, and optical sensors.

The strain of a beam element is a direct indication of curvature, which is the second spatial derivative of beam deflection, as indicated in (13.55). Modern semiconductor strain gauges have high gauge factors producing a good signal-to-noise ratio. The second spatial derivative relationship means that a slight change in location can result in a large change in the reading, but once affixed to the arm this is not a problem. More of a problem is the relatively short lifetime of a strain gauge subject to wear and tear if not carefully protected. By suitably placing strain gauges, good measurements of multiple assumed modes are possible.

Optical measurement of deflection can be very versatile but is subject to interference between a target measurement point and the sensor, particularly if the sensor is stationary. If the sensor is mounted on the link this difficulty is reduced but optical measurements are then usually a complex combination of the rotation of the optical axis of the sensor and the deflection of the target. Machine vision senses a wide field of view but typically with less precision. Multiple points can be detected, which is attractive in measuring multiple modes. Machine vision requires considerable processing and, even for enhanced targets or retroreflective fiducials, sample rates can be slow relative to some modes of control interest.

Accelerometers are attractive transducers in that they do not require explicit grounding to a fixed frame. Microelectromechanical systems (MEMS) accelerometers provide the required low-frequency sensitivity and are very inexpensive. However, they are subject to orientation in a gravity field, which must be compensated for by knowing the orientation of the sensor. If one is

interested in position, accelerometers require two integrations and are consequently subject to drift in position measurement. Consequently the combined use of sensors is appealing. For example, vision sensors, which are relatively slow but give direct position information for the end of a flexible member, can be combined with accelerometers, which give information about the same point at high sample rates but with noise and drift problems. The two sensor readings can be combined with a Kalman filter or other fusion scheme.

Issues Related to Model Order

Realizing that the theoretical order of a flexible arm is infinite, one should be prepared to deal with certain issues related to model order. The first of these is possible aliasing problems arising from finite sampling rates. Anti-aliasing filters should be employed, and even more effective is the use of analog damping by materials, damping treatments, or the back electromotive force (emf) of actuators to provide an inherent damping of the system under digital control. A more subtle problem is that of measurement and control spill over.

Control of Special Configurations: Macro/Microredundancy

Several special configurations have been proposed to enable improved controlled performance of flexible link arms. The problem with noncollocation noted above is a case in point. In some cases a large range of motion and high accuracy cannot be achieved with a single set of actuators. Redundant degrees of freedom can then be employed with the macro degrees of freedom that provide large range of motion but with inherent lack of precision or accuracy. Vibrations are particularly difficult to eliminate with a long arm, and construction accuracy and even thermal drift can exceed the tolerance limits. If the gross motion is completed quickly, residual vibrations may remain and be very lightly damped. Actuators constructed for gross motions of large axes do not have the bandwidth capabilities to damp these vibrations actively, but small motion axes typically do. This has been illustrated in several instances for long-reach manipulators when exploring nuclear waste cleanup solutions [13.61–64]. Using the inertial forces generated at the end of the arm also has the advantage of creating forces collocated with the measurements being used for control, assuming the base of the micro arm is at the end of the macro arm and the location of sensors. Loper [13.63] and George [13.65] have shown that this control is possible based on accelerometers in multiple degrees of freedom. It is also possible to excite unde-

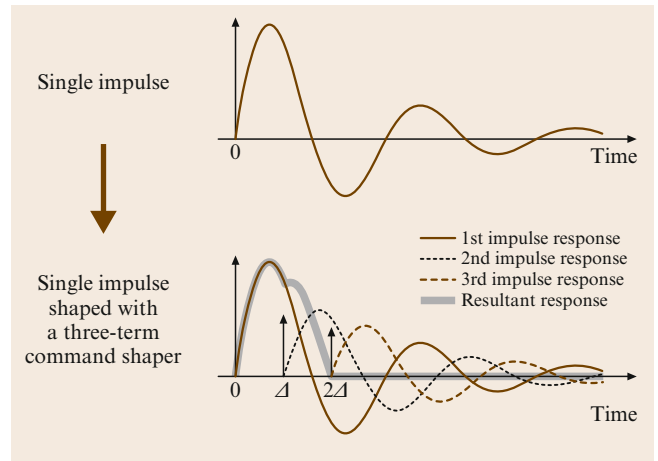


Fig. 13.11 Impulse response of a shaper and the effect on a vibratory system. Time is normalized by the intersample time T_s and T_d , chosen such that $\Delta = T_d/T_s$ is an integer

sired modes which have not been accounted for. If the full six degrees of freedom are used it is possible to argue stability based on a simple passive analogy. Generating prescribed forces at the base of an articulated micromanipulator is a challenging task, requiring the solution of a new type of inverse dynamics problem, which solves for motions or joint torques to produce the prescribed base forces and moments.

Command Generation for Flexible Link Arms

The trajectory undertaken by a flexible arm can dramatically affect the consequences of flexibility. Purely open-loop anticipation of the flexible dynamics has been used to create motion profiles wherein the motion cancels the incipient oscillation that had been created by earlier motions. This strategy is referred to as command shaping. A more complicated strategy to implement is a true inverse-dynamics-based trajectory wherein the

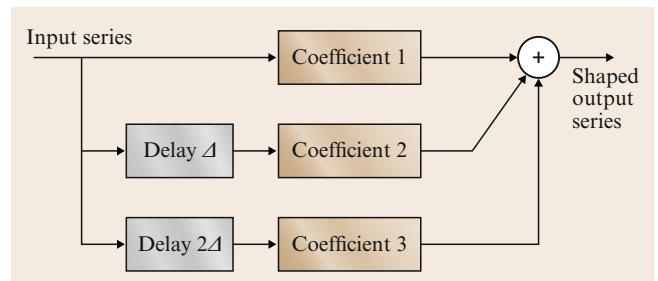


Fig. 13.12 Block diagram representation of the command shaping algorithm

rigid and flexible states are all orchestrated to arrive at the desired point in state space, typically at a static equilibrium. The high order of the flexible model is further complicated by the non-minimum-phase nature of the non-collocated system and noncausal inverse is necessary to produce a physically realizable trajectory.

Time delay command shaping is a variation of the deadbeat control introduced by [13.66, 67]. It can be couched in terms of a finite impulse response filter with a simple impulse response as depicted in Fig. 13.11 with a single input impulse producing a small number (typically two to four) of output impulses with appropriate time spacing and amplitudes. Extensions by *Singer* and *Seering* [13.68] and *Singhose* [13.69] have produced more-robust varieties under the trademark Input Shaping. The robustness is enhanced by using more output impulses and appropriate selection of the impulse spacing. The optimal arbitrary time-delay

(OAT) filter [13.70, 71] reduces this selection to a simple formula for linear vibrations that depends on the natural frequency and damping ratio of those modes to be eliminated from the response. The simplicity of implementation is illustrated in Fig. 13.12, where the relationships between the filter parameters and the mode parameters are

$$\text{coefficient 1} = 1/M$$

$$\text{coefficient 2} = -(2 \cos \omega_d T_d e^{-\zeta \omega_n T_d})/M$$

$$\text{coefficient 3} = (e^{-2\zeta \omega_n T_d})/M$$

$$M = 1 - 2 \cos \omega_d T_d e^{-\zeta \omega_n T_d} + e^{-2\zeta \omega_n T_d}$$

$$\omega_d = \omega_n \sqrt{1 - \zeta^2} = \text{damped natural frequency}$$

$$\omega_n = \text{undamped natural frequency}$$

$$\zeta = \text{damping ratio}$$

$$T_d = \text{time delay selected, an integer number of samples} = \Delta \quad (13.82)$$

Note that the formulation here is oriented towards digital systems with fixed sampling intervals such that the time delay of the filter is adjusted to an integer number of sampling intervals. Command shaping has been applied to a wide range of flexible systems including disk drives, cranes, and large robot arms, such as the example by CAMotion shown in Fig. 13.13. Note that this example contains a flexible drive belt, effectively a flexible joint, as well as flexible links that contribute to the compliance of the lowest mode that is quenched with an OAT filter.

Adaptive versions of command shaping have also been demonstrated. A version oriented towards repetitive motion applications is presented in [13.72].

In contrast to the scant system knowledge required for command shaping, the inverse dynamics of a flexible arm requires a great deal of knowledge about the system. The creation of an appropriate model was discussed above. An assumed modes model, for example, is well adapted to inverse dynamics if the rigid coordinates are not the joint angles, but the angles between lines connecting the joint axes (where it is assumed that joints are rotational). In this case the desired end-of-arm motion can be expressed solely in terms of rigid variables, that is, the same variables \mathbf{q}_d used in the flexible joint case, and the desired flexible variables are found from the second-order equations driven by the known rigid positions and velocities. The re-



Fig. 13.13 A large flexible linking arm with command shaping

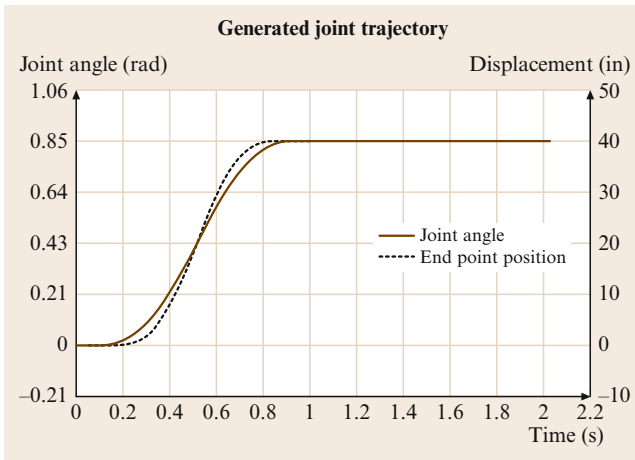


Fig. 13.14 Inverse dynamics driven motion of a single-link flexible arm

maining difficulty is the unstable zero dynamics of the system.

For the purposes of introduction, only the linear situation will be treated here in the fashion of [13.73]. First, the rigid coordinates and their derivatives are found from the desired end-of-arm motion profile. The flexible equations are separated into causal stable zeros and anticausal unstable zeros, and the stable inverse is solved forward in time. Physically the motion response responds after the torque inputs. The computational input however is the response and the torque is calculated as an output, so no physical laws are violated in solving the anticausal portion beginning at the final time when flexible coordinates are specified to be zero and proceeding to the initial time and beyond. In fact the exact solution extends from negative infinity (where the anticausal solution decays) to positive infinity (where the causal solution decays). For realistic dynamics the necessary time for the input is only slightly more than the motion time for the end of the arm because the zeros are often quite far from the origin in both directions. If the end-of-arm motion begins at time zero, shortly before that, the joint torques begin to *pre-shape* the flexible links without moving the end of arm. Similarly, when the end of the arm reaches its final position, the joint torques must release the arm deflection in a manner that is not seen from the end of arm (or any other chosen output location). Figure 13.14 shows the example from Kwon [13.74] for a single-link arm.

Feedback Control of Flexible Links

Flexibility in the links is in some cases controlled in the same way as concentrated flexibility in joints. Herein the focus is on approaches that apply specifically to link flexibility; consequently the use of Kalman filtering to estimate tip position and other broadly used techniques are not reexamined. A long history of research on this topic exists. Cannon and Schmitz [13.75], Truckenbrodt [13.76], and others showed that these techniques work for flexible link arms and that the non-collocated nature of the system aggravates the sensitivity to modeling errors.

Strain measurements provide a valuable indication of the state of a flexible link, and strain rate, obtained from filtered differentiation of the typically noisy strain signal, was shown by Hastings [13.77] to be an appropriate way of overcoming the limitations imposed by purely joint feedback control. Yuan [13.78] produced an adaptive version of strain-rate feedback.

Strain measurements readily lend themselves to calculation of the modal amplitudes, and the number of

appropriately placed strain gage measurements N can be equal to the number of mode amplitudes to be evaluated. In this case no estimation algorithm is necessary, although filtering is usually needed. If the mode shape ϕ_i of a beam is known, strain is proportional to the second derivative of ϕ_i ,

$$\alpha_{ij} = \left. \frac{\partial^2 \phi_i}{\partial x^2} \right|_{x_j} . \quad (13.83)$$

By inverting the coefficient matrix thus formed it is simple to obtain the amplitude of N modes. Additional attention is necessary if beam torsion is a significant part of the mode deflection, but the principle still applies. Active control of high-frequency modes may not be warranted, and feedback of the one or two lowest-frequency modes can be managed by classical or state-space design



Fig. 13.15 Robotic arm large and flexible (RALF) used in EOA feedback experiments

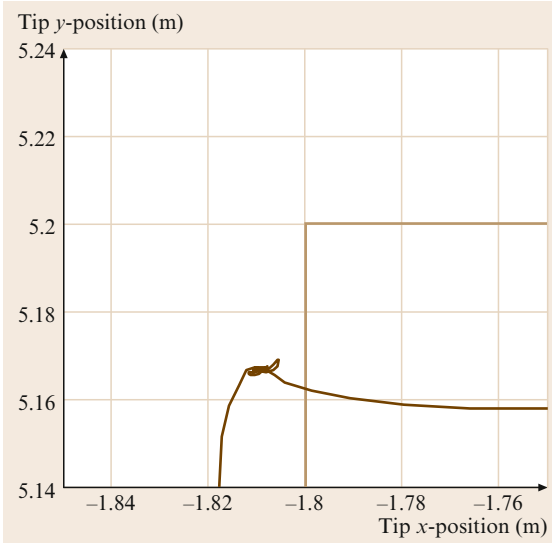


Fig. 13.16 Path followed without EOA position feedback. (Commanded trajectory is a *square* partially shown in light brown. Actual trajectory shown in dark brown.)

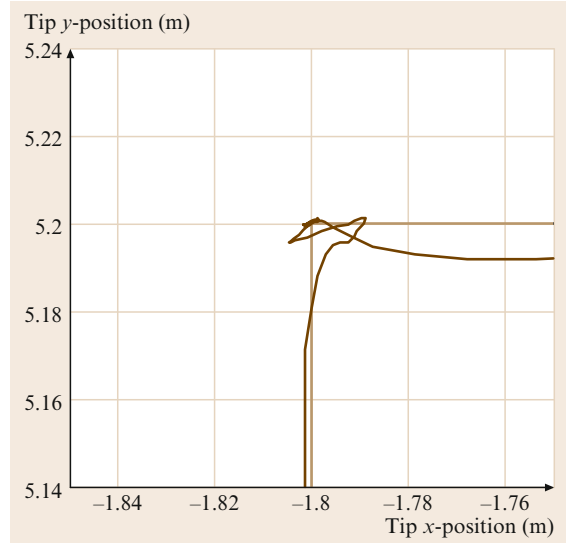


Fig. 13.17 Path followed with EOA feedback from camera. (Commanded trajectory is a *square* partially shown in light brown. Actual trajectory shown in dark brown.)

techniques to increase the damping of that mode

$$\begin{pmatrix} \delta_1 \\ \delta_2 \\ \vdots \\ \delta_N \end{pmatrix} = \begin{pmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1N} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{N1} & \alpha_{N2} & \dots & \alpha_{NN} \end{pmatrix}^{-1} \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}. \quad (13.84)$$

The use of the measurement of the end-of-arm position has perpetual appeal and is very effective for rigid arms for the elimination of variability due to arm distortion and the change over time of arm calibration. Machine vision can eliminate uncertainty in the target position as well as the arm position. These approaches are plagued by the non-collocated and hence non-minimum-phase dynamics of link flexibility, however, and may require particular caution. Wang [13.79] proposed an alternative formulation of EOA measurements in the control formulation that he called the reflected tip position. For a rotating link of angle θ and length l the deflected position is $l\theta + \delta$ while the reflected tip position $l\theta - \delta$ is minimum phase. This combination of joint and tip measurements produced the desired undeflected position but would not serve for EOA tracking control. Oberfell [13.80] synthesized an output based on strain feedback combined with EOA position that allowed tracking control as well. He first

produced a passive system with deflection feedback obtained from optical measurements of the link deflection, and then devised a traditional feedback control of the tip position measured by machine vision of the large two link arm shown in Fig. 13.15. Figures 13.16 and 13.17 compare the path for a large square traced in a plane with only the deflection feedback and with both deflection and EOA position feedback.

13.2.4 Further Reading

The topic of flexible link robot control is widely covered with papers on modeling and control. The reader wishing to probe further may want to begin with a paper giving some perspective on the nature of the problem. This was the goal of Book in [13.81]. Progress to date in this field from the theoretical perspective is significant and has been collected by Canudas de Witt et al. in [13.82]; this reference includes a wide range of research approaches, but the examples of the practical application of this theory are more limited. The German Aerospace Center has produced lightweight space arms [13.83] and deployed them in space. The startup CAMotion Inc. [13.84] has produced large lightweight industrial robots using the command-shaping approach to improve cycle time. High-speed arms such as the FlexPicker [13.85] achieve a light weight and avoid flexibility because of their parallel structure, but suffer

the usual disadvantages of parallel actuation: restricted workspace considering the space occupied by the robot structure.

Trautt and Bayo's early exploration of non-minimum-phase inverses is described in [13.86]. This is a particularly interesting variation of the work by Kwon [13.73, 74] in that it uses the frequency domain and deals with nonzero initial conditions. The concept of using two time scales, a fast time scale for the flexible dynamics and a slow time scale for the rigid dynamics, can be found in Siciliano and Book [13.87]. The singular per-

turbation approach was used earlier by Spong for elastic joint arms. Ghorbel and Spong also consider flexible links and contact with a rigid environment in [13.88]. Achieving robustness is a problem, particularly with end-point sensors. Sliding mode control [13.89] and passivity-based control [13.90] address this problem in effective but widely varying ways. While these seem to be the more promising methods of control, one will find many more approaches to this challenging control problem. H_∞ , adaptive, fuzzy, and other controllers have all been applied with varying degrees of success.

References

- 13.1 L.M. Sweet, M.C. Good: Redefinition of the robot motion control problem, *IEEE Control Syst. Mag.* **5**(3), 18–24 (1985)
- 13.2 M.C. Good, L.M. Sweet, K.L. Strobel: Dynamic models for control system design of integrated robot and drive systems, *ASME J. Dyn. Syst. Meas. Contr.* **107**, 53–59 (1985)
- 13.3 E. Rivin: *Mechanical Design of Robots* (McGraw-Hill, New York 1988)
- 13.4 S. Nicosia, F. Nicolò, D. Lentini: Dynamical control of industrial robots with elastic and dissipative joints, 8th IFAC World Congr. (Kyoto 1981) pp. 1933–1939
- 13.5 P. Tomei: An observer for flexible joint robots, *IEEE Trans. Autom. Control* **35**(6), 739–743 (1990)
- 13.6 R. Höppler, M. Thümmel: Symbolic computation of the inverse dynamics of elastic joint robots, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp. 4314–4319
- 13.7 M.W. Spong: Modeling and control of elastic joint robots, *ASME J. Dyn. Syst. Meas. Control* **109**, 310–319 (1987)
- 13.8 S.H. Murphy, J.T. Wen, G.N. Saridis: Simulation and analysis of flexibly jointed manipulators, 29th IEEE Conf. Decis. Control (Honolulu 1990) pp. 545–550
- 13.9 R. Marino, S. Nicosia: *On the Feedback Control of Industrial Robots with Elastic Joints: A Singular Perturbation Approach* (Dipartimento di Ingegneria Elettronica, Univ. Rome Tor Vergata, Rome 1984), Rep. R-84.01
- 13.10 G. Cesareo, F. Nicolò, S. Nicosia: DYMER: A code for generating dynamic model of robots, *IEEE Int. Conf. Robot. Autom.* (Atlanta 1984) pp. 115–120
- 13.11 A. De Luca: Feedforward/feedback laws for the control of flexible robots, *IEEE Int. Conf. Robot. Autom.* (San Francisco 2000) pp. 233–240
- 13.12 H.B. Kuntze, A.H.K. Jacobasch: Control algorithms for stiffening an elastic industrial robot, *IEEE J. Robot. Autom.* **1**(2), 71–78 (1985)
- 13.13 S.H. Lin, S. Tosunoglu, D. Tesar: Control of a six-degree-of-freedom flexible industrial manipulator, *IEEE Control Syst. Mag.* **11**(2), 24–30 (1991)
- 13.14 P. Tomei: A simple PD controller for robots with elastic joints, *IEEE Trans. Autom. Control* **36**(10), 1208–1213 (1991)
- 13.15 A. De Luca, B. Siciliano: Regulation of flexible arms under gravity, *IEEE Trans. Robot. Autom.* **9**(4), 463–467 (1993)
- 13.16 A. De Luca, B. Siciliano, L. Zollo: PD control with on-line gravity compensation for robots with elastic joints: Theory and experiments, *Automatica* **41**(10), 1809–1819 (2005)
- 13.17 C. Ott, A. Albu-Schäffer, A. Kugi, S. Stramigioli, G. Hirzinger: A passivity based Cartesian impedance controller for flexible joint robots – Part I: Torque feedback and gravity compensation, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp. 2659–2665
- 13.18 L. Zollo, B. Siciliano, A. De Luca, E. Guglielmelli, P. Dario: Compliance control for an anthropomorphic robot with elastic joints: Theory and experiments, *ASME J. Dyn. Syst. Meas. Control* **127**(3), 321–328 (2005)
- 13.19 A. Albu-Schäffer, C. Ott, G. Hirzinger: A passivity based Cartesian impedance controller for flexible joint robots – Part II: Full state feedback, impedance design and experiments, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp. 2666–2672
- 13.20 R. Kelly, V. Santibanez: Global regulation of elastic joint robots based on energy shaping, *IEEE Trans. Autom. Control* **43**(10), 1451–1456 (1998)
- 13.21 J. Alvarez-Ramirez, I. Cervantes: PID regulation of robot manipulators with elastic joints, *Asian J. Control* **5**(1), 32–38 (2003)
- 13.22 A. De Luca, S. Panzieri: Learning gravity compensation in robots: Rigid arms, elastic joints, flexible links, *Int. J. Adapt. Contr. Signal Process.* **7**(5), 417–433 (1993)
- 13.23 A. Albu-Schäffer, G. Hirzinger: A globally stable state feedback controller for flexible joint robots, *Adv. Robot.* **15**(8), 799–814 (2001)
- 13.24 L.E. Pfeffer, O. Khatib, J. Hake: Joint torque sensory feedback in the control of a PUMA manipulator, *IEEE Trans. Robot. Autom.* **5**(4), 418–425 (1989)

- 13.25 M. Hashimoto, Y. Kiyosawa, R.P. Paul: A torque sensing technique for robots with harmonic drives, *IEEE Trans. Robot. Autom.* **9**(1), 108–116 (1993)
- 13.26 T. Lin, A.A. Goldenberg: Robust adaptive control of flexible joint robots with joint torque feedback, *IEEE Int. Conf. Robot. Autom.* (Nagoya 1995) pp. 1229–1234
- 13.27 M.G. Forrest–Barlach, S.M. Babcock: Inverse dynamics position control of a compliant manipulator, *IEEE J. Robot. Autom.* **3**(1), 75–83 (1987)
- 13.28 K.P. Jankowski, H. Van Brussel: An approach to discrete inverse dynamics control of flexible-joint robots, *IEEE Trans. Robot. Autom.* **8**(5), 651–658 (1992)
- 13.29 W.M. Grimm: Robustness analysis of nonlinear decoupling for elastic-joint robots, *IEEE Trans. Robot. Autom.* **6**(3), 373–377 (1990)
- 13.30 A. De Luca, R. Farina, P. Lucibello: On the control of robots with visco-elastic joints, *IEEE Int. Conf. Robot. Autom.* (Barcelona 2005) pp. 4297–4302
- 13.31 A. De Luca: Decoupling and feedback linearization of robots with mixed rigid/elastic joints, *Int. J. Robust Nonlin. Contr.* **8**(11), 965–977 (1998)
- 13.32 A. De Luca, B. Siciliano: Inversion-based nonlinear control of robot arms with flexible links, *AIAA J. Guid. Control Dyn.* **16**(6), 1169–1176 (1993)
- 13.33 A. De Luca: Dynamic control of robots with joint elasticity, *IEEE Int. Conf. Robot. Autom.* (Philadelphia 1988) pp. 152–158
- 13.34 S. Nicosia, P. Tomei: On the feedback linearization of robots with elastic joints, 27th IEEE Conf. Decis. Control (Austin 1988) pp. 180–185
- 13.35 A. De Luca, P. Lucibello: A general algorithm for dynamic feedback linearization of robots with elastic joints, *IEEE Int. Conf. Robot. Autom.* (Leuven 1998) pp. 504–510
- 13.36 K. Khorasani, P.V. Kokotovic: Feedback linearization of a flexible manipulator near its rigid body manifold, *Syst. Control Lett.* **6**, 187–192 (1985)
- 13.37 M.W. Spong, K. Khorasani, P.V. Kokotovic: An integral manifold approach to the feedback control of flexible joint robots, *IEEE J. Robot. Autom.* **3**(4), 291–300 (1987)
- 13.38 S. Nicosia, P. Tomei: Design of global tracking controllers for flexible-joint robots, *J. Robot. Syst.* **10**(6), 835–846 (1993)
- 13.39 B. Brogliato, R. Ortega, R. Lozano: Global tracking controllers for flexible-joint manipulators: A comparative study, *Automatica* **31**(7), 941–956 (1995)
- 13.40 M.W. Spong: Adaptive control of flexible joint manipulators, *Syst. Control Lett.* **13**(1), 15–21 (1989)
- 13.41 R. Lozano, B. Brogliato: Adaptive control of robot manipulators with flexible joints, *IEEE Trans. Autom. Control* **37**(2), 174–181 (1992)
- 13.42 H. Sira-Ramirez, M.W. Spong: Variable structure control of flexible joint manipulators, *Int. Robot. Autom.* **3**(2), 57–64 (1988)
- 13.43 A. De Luca, G. Ulivi: Iterative learning control of robots with elastic joints, *IEEE Int. Conf. Robot. Autom.* (Nice 1992) pp. 1920–1926
- 13.44 S. Nicosia, P. Tomei, A. Tornambè: A nonlinear observer for elastic robots, *IEEE J. Robot. Autom.* **4**(1), 45–52 (1988)
- 13.45 S. Nicosia, P. Tomei: A method for the state estimation of elastic joint robots by global position measurements, *Int. J. Adapt. Contr. Signal Process.* **4**(6), 475–486 (1990)
- 13.46 A. De Luca, D. Schröder, M. Thümmel: An acceleration-based state observer for robot manipulators with elastic joints, *IEEE Int. Conf. Robot. Autom.* (Rome 2007) pp. 3817–3823
- 13.47 M.W. Spong: On the force control problem for flexible joint manipulators, *IEEE Trans. Autom. Control* **34**(1), 107–111 (1989)
- 13.48 J.K. Mills: Stability and control of elastic-joint robotic manipulators during constrained-motion tasks, *IEEE Trans. Robot. Autom.* **8**(1), 119–126 (1992)
- 13.49 K.P. Jankowski, H.A. El Maraghy: Dynamic decoupling for hybrid control of rigid-/flexible-joint robots interacting with the environment, *IEEE Trans. Robot. Autom.* **8**(5), 519–534 (1992)
- 13.50 T. Lin, A.A. Goldenberg: A unified approach to motion and force control of flexible joint robots, *IEEE Int. Conf. Robot. Autom.* (Minneapolis 1996) pp. 1115–1120
- 13.51 W. Book, O. Maizza-Neto, D.E. Whitney: Feedback Control of Two Beam, Two Joint Systems with Distributed Flexibility, *ASME J. Dyn. Syst. Meas. Control* **97**(4), 424–431 (1975)
- 13.52 W. Book: Characterization of Strength and Stiffness Constraints on Manipulator Control, In: *Theory and Practice of Robots and Manipulators* (Elsevier, Amsterdam 1977) pp. 37–45
- 13.53 T.E. Alberts, W. Book, S. Dickerson: Experiments in Augmenting Active Control of a Flexible Structure with Passive Damping, *AIAA 24th Aerospace Sciences Meeting* (Reno 1986)
- 13.54 T. Bailey, J.E. Hubbard Jr.: Distributed piezoelectric-polymer active vibration control of a cantilever beam, *J. Guid. Control Dyn.* **8**(5), 605–611 (1985)
- 13.55 W. Book, V. Sangveraphunsiri, S. Le: The Bracing Strategy for Robot Operation, Joint IFTOMM-CISM Symposium on the Theory of Robots and Manipulators (RoManSy) (Udine 1984)
- 13.56 W. Book: Analysis of Massless Elastic Chains with Servo Controlled Joints, *ASME J. Dyn. Syst. Meas. Control* **101**(3), 187–192 (1979)
- 13.57 E.C. Pestel, F.A. Leckie: *Matrix Methods in Elastomechanics* (McGraw Hill, New York 1963)
- 13.58 R. Krauss: Transfer Matrix Modeling. Ph.D. Thesis (School of Mechanical Engineering, Georgia Institute of Technology 2006)
- 13.59 W.J. Book: Modeling, Design and Control of Flexible Manipulator Arms. Ph.D. Thesis (Department of

- Mechanical Engineering, Massachusetts Institute of Technology 1974)
- 13.60 W. Book: Recursive Lagrangian Dynamics of Flexible Manipulators, *Int. J. Robot. Res.* **3**(3), 87–106 (1984)
- 13.61 W.J. Book, S.H. Lee: Vibration Control of a Large Flexible Manipulator by a Small Robotic Arm, *Proc. Am. Contr. Conf.* (Pittsburgh 1989) pp. 1377–1380
- 13.62 J. Lew, S.-M. Moon: A Simple Active Damping Control for Compliant Base Manipulators, *IEEE/ASME Trans. Mechatronics* **2**, 707–714 (1995)
- 13.63 W.J. Book, J.C. Loper: Inverse Dynamics for Commanding Micromanipulator Inertial Forces to Damp Macromanipulator Vibration, *IEEE, Robot Society of Japan International Conference on Intelligent Robots and Systems* (Kyongju 1999)
- 13.64 I. Sharf: Active Damping of a Large Flexible Manipulator with a Short-Reach Robot, *Proc. Of the American Control Conference* (Seattle 1995) pp. 3329–3333
- 13.65 L. George, W.J. Book: Inertial Vibration Damping Control of a Flexible Base Manipulator, *IEEE/ASME Transactions on Mechatronics* (2003)
- 13.66 J.F. Calvert, D.J. Gimpel: Method and Apparatus for Control of System Output in Response to System Input, Patent 2801351 (1957)
- 13.67 O.J.M. Smith: *Feedback Control Systems* (McGraw Hill, New York 1958)
- 13.68 N. Singer, W.P. Seering: Preshaping Command Inputs to Reduce System Vibration, *ASME J. Dyn. Syst. Meas. Contr.* **112**(1), 76–82 (1990)
- 13.69 W. Singhose, W. Seering, N. Singer: Residual Vibration Reduction Using Vector Diagrams to Generate Shaped Inputs, *J. Mech. Des.* **2**, 654–659 (1994)
- 13.70 D.P. Magee, W.J. Book: The Application of Input Shaping to a System with Varying Parameters, *Proceedings of the 1992 Japan-U.S.A. Symposium on Flexible Automation* (San Francisco 1992) pp. 519–526
- 13.71 D.P. Magee, W.J. Book: Optimal Arbitrary Time-delay (OAT) Filter and Method to Minimize Unwanted System Dynamics, Patent 6078844 (2000)
- 13.72 S. Rhim, W.J. Book: Noise Effect on Time-domain Adaptive Command Shaping Methods for Flexible Manipulator Control, *IEEE Trans. Control Syst. Technol.* **9**(1), 84–92 (2001)
- 13.73 D.-S. Kwon, W.J. Book: A Time-Domain Inverse Dynamic Tracking Control of a Single-Link Flexible Manipulator, *J. Dyn. Syst. Meas. Control* **116**, 193–200 (1994)
- 13.74 D.S. Kwon: An Inverse Dynamic Tracking Control for a Bracing Flexible Manipulator. Ph.D. Thesis (School of Mechanical Engineering, Georgia Institute of Technology 1991)
- 13.75 R.H. Cannon, E. Schmitz: Initial Experiments on the End-Point Control of a Flexible One-Link Robot, *Int. J. Robot. Res.* **3**(3), 62–75 (1984)
- 13.76 A. Truckenbrot: Modeling and Control of Flexible Manipulator Structures, *Proc. 4th CISM-IFTOMM Ro.Man.Sy.* (Zaborow 1981) pp. 90–101
- 13.77 G.G. Hastings, W.J. Book: Reconstruction and Robust Reduced-Order Observation of Flexible Variables, *ASME Winter Annual Meeting* (Anaheim 1986)
- 13.78 B.S. Yuan, J.D. Huggins, W.J. Book: Small Motion Experiments with a Large Flexible Arm with Strain Feedback, *Proceedings of the 1989 American Control Conference* (Pittsburgh 1989) pp. 2091–2095
- 13.79 D. Wang, M. Vidyasagar: Passive Control of a Stiff Flexible Link, *Int. J. Robot. Res.* **11**, 572–578 (1992)
- 13.80 K. Oberfell, W.J. Book: Control of Flexible Manipulators Using Vision and Modal Feedback, *Proceedings of the ICRAM* (Istanbul 1995)
- 13.81 W.J. Book: Controlled Motion in an Elastic World, *ASME J. Dyn. Syst. Meas. Control* **2B**, 252–261 (1993), 50th Anniversary Issue
- 13.82 C. Canudas-de-Wit, B. Siciliano, G. Bastin (eds.): *Theory of Robot Control* (Springer, Berlin, Heidelberg 1996)
- 13.83 G. Hirzinger, N. Sporer, J. Butterfass, M. Grebenstein: Torque-Controlled Lightweight Arms and Articulated Hands: Do We Reach Technological Limits Now?, *Int. J. Robot. Res.* **23**(4,5), 331–340 (2004)
- 13.84 Camotion Inc.: <http://www.camotion.com> (Camotion Inc., Atlanta 2007)
- 13.85 B. Rooks: High Speed Delivery and Low Cost from New ABB Packaging Robot, *Ind. Robot. Int. J.* **26**(4), 267–275 (1999)
- 13.86 T. Trautt, E. Bayo: Inverse Dynamics of Non-Minimum Phase Systems with Non-Zero Initial Conditions, *Dyn. Control* **7**(1), 49–71 (1997)
- 13.87 B. Siciliano, W. Book: A Singular Perturbation Approach to Control of Lightweight Flexible Manipulators, *Int. J. Robot. Res.* **7**(4), 79–90 (1988)
- 13.88 F. Ghorbel, M.W. Spong: Singular Perturbation Model of Robots with Elastic Joints and Elastic Links Constrained by a Rigid Environment, *J. Intell. Robot. Syst.* **22**(2), 143–152 (1998)
- 13.89 J. Guldner, J. Shi, V. Utkin: *Sliding Mode Control in Electromechanical Systems* (Taylor Francis, London, Philadelphia 1999)
- 13.90 J.-H. Ryu, D.-S. Kwon, B. Hannaford: Control of a Flexible Manipulator with Noncollocated Feedback: Time-domain Passivity Approach, *IEEE Trans. Robot.* **20**(4), 776–780 (2004)

Model Identi

14. Model Identification

John Hollerbach, Wisama Khalil, Maxime Gautier

This chapter discusses how to determine the kinematic parameters and the inertial parameters of robot manipulators. Both instances of model identification are cast into a common framework of least-squares parameter estimation, and are shown to have common numerical issues relating to the identifiability of parameters, adequacy of the measurement sets, and numerical robustness. These discussions are generic to any parameter estimation problem, and can be applied in other contexts.

For kinematic calibration, the main aim is to identify the geometric Denavit–Hartenberg (DH) parameters, although joint-based parameters relating to the sensing and transmission elements can also be identified. Endpoint sensing or endpoint constraints can provide equivalent calibration equations. By casting all calibration methods as closed-loop calibration, the calibration index categorizes methods in terms of how many equations per pose are generated.

Inertial parameters may be estimated through the execution of a trajectory while sensing one or more components of force/torque at a joint. Load estimation of a handheld object is simplest because of full mobility and full wrist force–torque sensing. For link inertial parameter estimation,

14.1 Overview.....	321
14.2 Kinematic Calibration	323
14.2.1 Serial-Link Robot Manipulators.....	323
14.2.2 Parallel Manipulator Calibration.....	328
14.3 Inertial Parameter Estimation.....	330
14.3.1 Load Inertial Parameter Estimation	330
14.3.2 Link Inertial Parameter Estimation .	332
14.3.3 Link Parameter Estimation for More Complex Structures	333
14.4 Identifiability and Numerical Conditioning	334
14.4.1 Identifiability	334
14.4.2 Observability	337
14.4.3 Scaling.....	339
14.5 Conclusions and Further Reading.....	341
14.5.1 Relation to Other Chapters	341
14.5.2 Further Reading.....	342
References	342

restricted mobility of links nearer the base as well as sensing only the joint torque means that not all inertial parameters can be identified. Those that can be identified are those that affect joint torque, although they may appear in complicated linear combinations.

14.1 Overview

There are many different kinds of models in robotics, whose accurate identification is required for precise control. Examples from the previous chapters include sensor models, actuator models, kinematic models, dynamic models, and flexibility models. System identification is the general field concerned with the process of identifying models from measurements. Generally speaking, there are two types of models: parametric and non-

parametric models. Parametric models are described by a few parameters, which are adequate to characterize the accuracy of a model throughout the working range. Examples include sensor gain and offset, the DH parameters for links, and rigid-body inertial parameters. Parametric models are particularly appropriate for robotics, whose components are manmade and whose properties are controlled and understood.

Nonparametric models include the impulse response and the Bode plot for linear systems, and Wiener and Volterra kernels for nonlinear systems [14.1]. A nonparametric model can be used as a stepping stone towards identifying a parametric model; for example, a Bode plot (graph of amplitude and phase versus input frequency) is often used to decide on model order, such as whether an actuator should be modeled as a second- or third-order system. Otherwise, nonparametric models are required when the system's properties are so complicated that a few lumped parameters do not suffice. This is particularly true for biological systems.

This chapter will describe the parametric calibration of the following kinds of models.

1. Kinematic parameters. Kinematic calibration is the process of locating coordinate systems of objects relative to each other. These objects may be isolated from each other, or they may be linked by joints. Examples include
 - locating a robot relative to a global reference frame
 - locating a stereo vision system relative to a robot
 - locating a grasped object relative to the manipulator's gripper frame
 - locating neighboring coordinate systems of links of a robot manipulator
2. Rigid-body inertial parameters. These parameters are required to predict the driving forces and torques to move objects and manipulators.

Suppose there are N_{par} parameters combined into an $N_{\text{par}} \times 1$ parameter vector $\phi = \{\phi_1, \dots, \phi_{N_{\text{par}}}\}$. The parameters may appear linearly or nonlinearly in the model.

$$\text{Linear model:} \quad y^l = A^l \phi, \quad (14.1)$$

$$\text{Nonlinear model:} \quad y^l = f(x^l, \phi), \quad (14.2)$$

where $y^l = \{y_1^l, \dots, y_M^l\}$ is the $M \times 1$ vector of output variables and $x^l = \{x_1^l, \dots, x_n^l\}$ is the vector of input variables. For the linear model, A^l is an $M \times N_{\text{par}}$ matrix whose entries A_{ij}^l are functions of the input variables x^l . Any entry A_{ij}^l may be a complicated nonlinear function of x^l , but it evaluates to just a number. For the nonlinear model, an explicit equation is shown in which the input variables appear in a nonlinear function $f = \{f_1, \dots, f_M\}$. Implicit nonlinear models $f(y^l, x^l, \phi) = 0$ may also appear in calibration [14.2]; they are handled similarly to explicit nonlinear models (Sect. 14.2.2). There may be P different measurements;

a particular measurement is indicated by the superscript $l = 1, \dots, P$.

For linear models, information from different measurements is combined by stacking the P equations (14.1):

$$y = A\phi, \quad (14.3)$$

where $y = \{y^1, \dots, y^P\}$ is an $MP \times 1$ vector of all output measurements and $A = \{A^1, \dots, A^P\}$ is $MP \times N_{\text{par}}$. The parameters are estimated by ordinary least squares:

$$\phi = (A^T A)^{-1} A^T y. \quad (14.4)$$

In statistics, the matrix A is called the regressor matrix and the least-squares solution is called regression [14.3]. An example of a linear model is the rigid-body model for inertial parameters.

The Gauss–Newton method [14.3] is typically employed to estimate the nonlinear model (14.2). First, absorb the input variables x^l (which can be considered as a number of constants) into the nonlinear function f^l , now given a superscript. The model is linearized through a Taylor series expansion at a current estimate ϕ^k of the parameters at iteration k :

$$\begin{aligned} y_c^l &= f^l(\phi^k + \Delta\phi) \\ &= f^l(\phi^k) + \left. \frac{\partial f^l(\phi)}{\partial \phi} \right|_{\phi=\phi^k} \Delta\phi \\ &\quad + \text{higher order terms} \\ &\approx f^l(\phi^k) + A^l \Delta\phi, \end{aligned} \quad (14.5)$$

where y_c^l are the computed values of the output variables and $A^l = \partial f^l / \partial \phi$ is a Jacobian matrix evaluated at ϕ^k . Higher-order terms in the Taylor series are ignored, yielding the linearized form (14.5). The bold assumption is now made that a correction $\Delta\phi$ to the parameter estimate ϕ^k causes the computed output variables to equal the measurements: $y_c^l = y^l$. Defining $\Delta y^l = y^l - f^l(\phi^k)$ as the error between the output measurement and the predicted output with the current model ϕ^k , the linearized equation (14.5) becomes

$$\Delta y^l = A^l \Delta\phi. \quad (14.6)$$

The linearized equation is then stacked for P measurements for the estimation form:

$$\Delta y = A \Delta\phi. \quad (14.7)$$

A correction $\Delta\phi$ to the parameter estimates is now found by ordinary least squares

$$\Delta\phi = (A^T A)^{-1} A^T \Delta y. \quad (14.8)$$

This process is iterated with a new estimate $\phi^{k+1} = \phi^k + \Delta\phi$ until the error $\Delta\phi$ becomes sufficiently small. The Gauss–Newton method has quadratic convergence, which is very fast, provided that there is a good initial estimate ϕ^0 of the parameters and the nonlinearity is not too severe. An example of a nonlinear model is the kinematic model containing the DH parameters. The nonlinearity is mild because it is due to sines and cosines, so the Gauss–Newton method usually has good convergence properties.

Both for linear and nonlinear estimation, however, rank deficiencies and numerical ill-conditioning may cause problems in inverting $A^T A$. Rank deficiencies may result from two problems.

1. Inadequate data. The quality of the data for estimating the parameters can be quantified by an observability index, such as the condition number [14.4] of the regressor matrix A . A different choice of data to maximize the observability index might result in more robust estimates. Examples are choosing different poses for kinematic calibration, or different trajectories for inertial parameter estimation.
2. Unidentifiable parameters. Perhaps no set of data from an experiment can identify some of the parameters. A procedure has to be found to eliminate or circumvent the unidentifiable parameters. Parameter elimination is usually done by using the singular value decomposition of A , while circumvention can be achieved by using a priori values and ridge regression. This does not mean that the parameters are intrinsically identifiable, only that the experimen-

tal setup precludes their determination; for example, only one of the 10 inertial parameters of the first link of a robot manipulator can be identified if the base of the manipulator is stationary. A different experimental setup involving accelerating the base of the manipulator and measuring the reaction forces and torques at the base would allow other parameters to be identified [14.5].

Ill-conditioning may result from poor scaling of measurements or parameters.

1. The least-squares estimation minimizes the error between the predicted and measured output vector. Components y_j^l of the output vector y^l may have different units and magnitudes, such as radians versus meters, in pose measurements for kinematic calibration. Moreover, not all components may be measured to the same level of accuracy. Normalizing the output vectors with an appropriately chosen weighting matrix may result in a better conditioned estimate.
2. The parameters may also have different units and different magnitudes. This can cause problems both in terms of convergence criteria and in terms of deciding which parameters to eliminate. Again, a weighting matrix for the parameters may be introduced to improve the conditioning.

These numerical issues are generic to any parameter estimation problem, and are discussed at the end of this chapter. The next sections discuss individual robot models and issues in putting them into parameter estimation form.

14.2 Kinematic Calibration

In general, the relative location between coordinate systems requires six geometric parameters to be calibrated (position plus orientation). If there are mechanical constraints between the relative movement of the coordinate systems, such as connection by a joint, fewer parameters are required. For a rotary joint connecting two links, whose axis is a line-bound vector, four geometric parameters are required. For a prismatic joint, whose axis is a free vector, only two geometric parameters describing orientation are required.

In addition, nongeometric parameters are required to model sensors and mechanical deflection.

- Joint angle sensors may require a gain and offset determination.
- Camera calibration using an undistorted pinhole camera model may require the determination of the focal length and an image sensor offset.
- Joint flexibility from gears leads to angle change due to gravity loading and the manipulator's own weight.
- Base flexibility results from nonrigid attachment of the robot to the environment. Depending on how the manipulator is outstretched, there will be a varying effect on endpoint location.
- Thermal effects and vibration may need to be modeled for very fine positioning control.

This section focuses on determining the geometric parameters and the sensor-based nongeometric parameters.

14.2.1 Serial-Link Robot Manipulators

The modified Denavit–Hartenberg (DH) parameters (Fig. 14.1) serve as the main geometric parameters (Sect. 1.4); the link transformation matrix is

$$\begin{aligned} {}^{i-1}T_i &= \text{Rot}(\mathbf{x}, \alpha_i) \text{Trans}(\mathbf{x}, a_i) \text{Trans}(\mathbf{z}, d_i) \text{Rot}(\mathbf{z}, \theta_i). \end{aligned} \quad (14.9)$$

- For an n -joint manipulator with rotary joint $i = 1, \dots, n$, whose axis \mathbf{z}_i is a line in space, all four parameters a_i , d_i , α_i , and θ_i have to be calibrated.
- For a prismatic joint i , whose axis \mathbf{z}_i is a free vector, only two parameters describing its orientation (α_i and θ_i) are required. In principle, the axis \mathbf{z}_i can be positioned anywhere in space. This means two DH parameters are arbitrary. One possibility is to intersect \mathbf{z}_i with O_{i+1} [14.6, 7], which sets $d_{i+1} = 0$ and $a_{i+1} = 0$. While kinematically correct, this placement is nonintuitive in that it doesn't correspond to the physical location of the prismatic mechanism. It is possible to set a_{i+1} with a_i or θ_i to values that position \mathbf{z}_i in the middle of the prismatic joint's mechanical structure.

In the case of nearly parallel neighboring axes, the common normal is poorly defined and the calibration is ill-conditioned. For this case, Hayati [14.7] introduced an extra rotational parameter β_i about the \mathbf{y}_{i-1} axis (Fig. 14.2).

Let \mathbf{x}'_{i-1} lie along a line from O_i to axis \mathbf{z}_{i-1} , such that \mathbf{x}'_{i-1} is perpendicular to \mathbf{z}_i ; the intersection point defines the origin O_{i-1} . Two rotations are required to

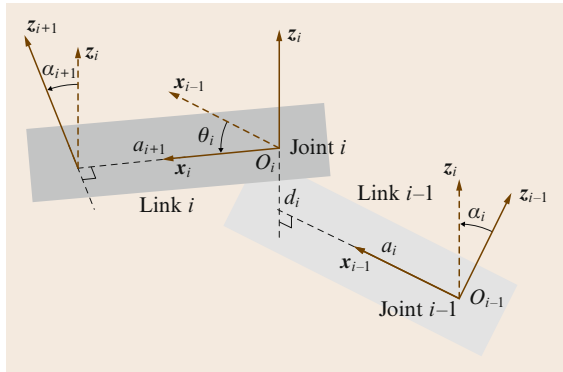


Fig. 14.1 The modified DH parameters

relate \mathbf{z}_{i-1} to $\mathbf{z}_i = \mathbf{z}'_{i-1}$: a rotation α_i about \mathbf{x}'_{i-1} maps \mathbf{z}'_{i-1} to \mathbf{z}_i , and a rotation β_i about $\mathbf{y}_{i-1} = \mathbf{y}''_{i-1}$ maps \mathbf{z}_{i-1} to \mathbf{z}'_{i-1} . The angle θ_i is now from \mathbf{x}'_{i-1} to \mathbf{x}_i about \mathbf{z}_i . The link transformation is

$$\begin{aligned} {}^{i-1}T_i &= \text{Rot}(\mathbf{y}, \beta_i) \text{Rot}(\mathbf{x}, \alpha_i) \text{Trans}(\mathbf{x}, a_i) \text{Rot}(\mathbf{z}, \theta_i). \end{aligned} \quad (14.10)$$

- For a rotary joint, the parameter β_i is calibrated instead of d_i .
- For a prismatic joint, the joint variable d_i has to be retained. As before, position \mathbf{z}_i at some convenient location on link i , by specifying two parameters relative to the coordinate system $i + 1$. Then proceed with the construction of the Hayati parameters for $d_i = 0$. As d_i changes, axis \mathbf{x}_i is displaced along \mathbf{z}_i relative to \mathbf{x}'_{i-1} (not shown). The link transformation is:

$$\begin{aligned} {}^{i-1}T_i &= \text{Rot}(\mathbf{y}, \beta_i) \text{Rot}(\mathbf{x}, \alpha_i) \text{Trans}(\mathbf{x}, a_i) \\ &\quad \text{Trans}(\mathbf{z}, d_i) \text{Rot}(\mathbf{z}, \theta_i). \end{aligned} \quad (14.11)$$

Although there are five parameters in this transformation, the process of setting the Hayati parameters for $d_i = 0$ is tantamount to setting the value of d_{i-1} to locate O_{i-1} , so there is no net increase in the number of parameters.

The procedures above set the coordinate systems in the intermediate links of a serial manipulator. Coordinate systems also have to be set in the base and end links, but the procedure for doing so depends on the external metrology system and on any physical constraints on the end link pose. The last frame may be n or $n + 1$, while the first frame may be 0 or -1 (to keep consecutive numbers); examples appear below. Collect the unknown kinematic parameters into the vectors \mathbf{a} , \mathbf{d} , $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$, and $\boldsymbol{\beta}$,

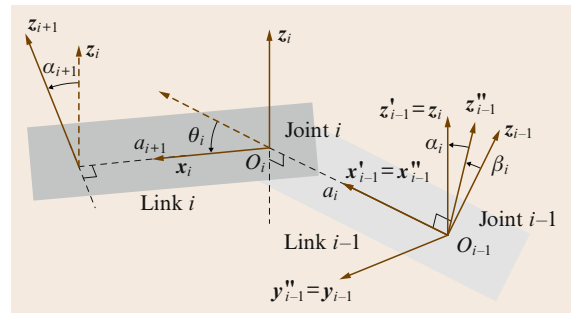


Fig. 14.2 The additional parameter β_i about \mathbf{y}_{i-1} is employed for nearly parallel axes

and thence into the parameter vector $\phi = \{a, d, \alpha, \theta, \beta\}$. The parameters ϕ predict the position and orientation of the last coordinate system relative to the first, such as ${}^0T_{n,c}$.

Not all six components of pose are necessarily used for calibration; the number can vary from one to six. Calibration proceeds by observing the error in the prediction of a certain number of pose components, then employing the nonlinear calibration method (14.7). There are two general methods for observing error.

- *Open-loop calibration* utilizes an external metrology system to measure the pose components. Because the manipulator is not in contact with the environment during this process, the method is termed open loop.
- *Closed-loop calibration* utilizes physical constraints on the end link pose to substitute for measurements. Deviations from the physical constraint represent the prediction error. Because of the contact with the physical constraint, the manipulator forms a closed loop with the ground.

Open-Loop Kinematic Calibration

The calibration literature contains a variety of metrology systems, which can be categorized based on the number of pose components measured [14.8].

- 1 component: The distance to a single point on the end link can be measured in a variety of ways, such as an instrumented ball bar [14.9], wire potentiometer [14.10], or laser displacement meter [14.11].
- 2 components: A single theodolite has been employed to provide two orientation measurements [14.12]. A reference length had to be sighted for scaling.
- 3 components: Laser tracking systems provide accurate three-dimensional (3-D) measurements by reflecting a beam off a retroreflector mounted on the end effector. The beam provides length information, while the gimbal drive for laser steering provides two orientation measurements [14.13]. Since the least precise part of this setup is the angle sensing, another approach is to employ three laser tracking systems and use only the length information. Commercial 3-D stereo-camera motion tracking systems also provide high-accuracy measurements of position.

5 components: Lau et al. [14.14] presented a steerable laser interferometer with steerable reflector. With pitch and yaw measurements, the steerable interferometer yields all three components of position, while the steerable reflector yields two components of orientation.

6 components: Full pose can be inferred from the 3-D position of multiple points on the last link measured with a stereo-camera system. A coordinate system fit to the cloud of points yields position plus orientation [14.15].

Vincze et al. [14.16] measured full pose with a single-beam laser tracking system, by fitting the robot with a retroreflector mounted on a universal joint. Position is measured using interferometry, as usual. The novel aspect is orientation measurement, by imaging of the diffraction pattern of the edges of the retroreflector.

Examples are given for calibrating with three pose components and with all six pose components measured.

Point Measurement. The 3-D position of a particular point on the end link can be conveniently located by some form of stereo camera system. The camera system defines a global coordinate system relative to which frame 1 of the robot is located. To provide enough parameters, an intermediate coordinate system has to be introduced; this intermediate coordinate system has index 0, while the camera system is given index -1 to keep consecutive numbers. Two of the eight parameters are arbitrary. Figure 14.3 shows one possibility: z_0 is made

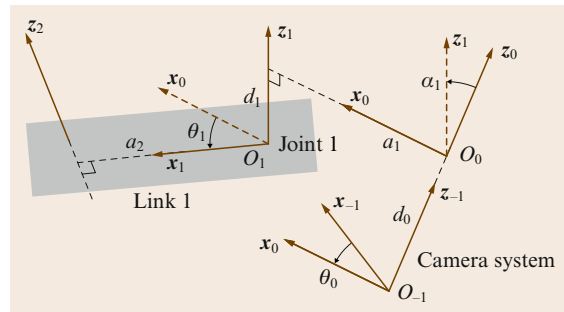


Fig. 14.3 A camera system (index -1) is located relative to the manipulator coordinate system 1 via an intermediate coordinate system 0

parallel ($\alpha_0 = 0$) and coincident ($a_0 = 0$) with z_{-1} . The calibrated parameters are $d_0, \theta_0, a_1, d_1, \alpha_1$, and θ_1 . In the case that z_{-1} is nearly parallel to z_0 , the measurement coordinate system can be simply redefined to avoid using Hayati parameters; for example, y_{-1} can be redefined as z_{-1} .

In the end link, the origin O_n and axis x_n are unspecified, as are the associated parameters d_n and θ_n . Locating the measured point requires only three parameters, so the addition of a single coordinate system $n+1$ is required to provide an additional parameter. The measured point is defined as the origin O_{n+1} of the new coordinate system, and the normal from z_n that intersects O_{n+1} defines the x_n axis (Fig. 14.4). Three parameters are arbitrary; a simple choice is to make z_{n+1} parallel to z_n ($\alpha_{n+1} = 0$) and x_{n+1} collinear with x_n ($\theta_{n+1} = 0$) and $d_{n+1} = 0$). The calibrated parameters are a_{n+1}, d_n , and θ_n .

From the transformation ${}^{-1}T_{n+1} = {}^{-1}T_0 \dots {}^nT_{n+1}$, the position ${}^{-1}p_{n+1}$ of the measured point relative to the camera frame is extracted. Collect the unknown kine-

matic parameters into vectors \mathbf{a} , \mathbf{d} , $\boldsymbol{\alpha}$, $\boldsymbol{\theta}$, and $\boldsymbol{\beta}$, and thence into the parameter vector $\boldsymbol{\phi} = \{\mathbf{a}, \mathbf{d}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{\beta}\}$. The nonlinear kinematic model analogous to (14.2) is

$${}^{-1}p_{n+1}^l = f(q^l, \boldsymbol{\phi}), \quad l = 1, \dots, P, \quad (14.12)$$

where q^l is the vector of joint variables for pose l . To linearize this equation into estimation form (14.6), the associated Jacobians for each parameter type are calculated:

$$\begin{aligned} \Delta^{-1}p_{n+1}^l &= {}^{-1}p_{n+1}^l - {}^{-1}p_{n+1,c}^l = \mathbf{J}^l \Delta \boldsymbol{\phi} \\ &= \begin{pmatrix} \mathbf{J}_a^l & \mathbf{J}_d^l & \mathbf{J}_\alpha^l & \mathbf{J}_\theta^l & \mathbf{J}_\beta^l \end{pmatrix} \begin{pmatrix} \Delta \mathbf{a} \\ \Delta \mathbf{d} \\ \Delta \boldsymbol{\alpha} \\ \Delta \boldsymbol{\theta} \\ \Delta \boldsymbol{\beta} \end{pmatrix}, \quad (14.13) \end{aligned}$$

where a typical column i for each individual parameter Jacobian is derived from the screw parameters as if each parameter represented an active joint (Sect. 1.8.1):

$$\mathbf{J}_{a_i}^l = \begin{Bmatrix} {}^{-1}x_{i-1}^l & \text{DH} \\ {}^{-1}x_{i-1}^l & \text{Hayati} \end{Bmatrix}, \quad (14.14)$$

$$\mathbf{J}_{d_i}^l = {}^{-1}z_i^l, \quad (14.15)$$

$$\mathbf{J}_{\alpha_i}^l = \begin{Bmatrix} {}^{-1}x_{i-1}^l \times {}^{-1}d_{i-1,n+1}^l & \text{DH} \\ {}^{-1}x_{i-1}^l \times {}^{-1}d_{i-1,n+1}^l & \text{Hayati} \end{Bmatrix}, \quad (14.16)$$

$$\mathbf{J}_{\theta_i}^l = {}^{-1}z_i^l \times {}^{-1}d_{i,n+1}^l, \quad (14.17)$$

$$\mathbf{J}_{\beta_i}^l = {}^{-1}y_{i-1}^l \times {}^{-1}d_{i-1,n+1}^l, \quad (14.18)$$

where ${}^{-1}d_{i,n+1}^l = {}^{-1}\mathbf{R}_i^l p_{n+1}$ refers the interorigin vector to coordinate system -1 . Equations (14.13) are stacked for all the poses for the final estimation form

$$\Delta^{-1}p_{n+1} = \mathbf{J} \Delta \boldsymbol{\phi}, \quad (14.19)$$

which corresponds to (14.7) and which is solved for $\Delta \boldsymbol{\phi}$ by least squares and for $\boldsymbol{\phi}$ by iteration.

Full Pose Measurement. Suppose that coordinate system $n+1$ in link n is measured (Fig. 14.5). Coordinate system n is completed in the usual way, and the six calibrated parameters to locate coordinate system $n+1$ are $d_n, \alpha_n, \theta_n, a_{n+1}, d_{n+1}$, and θ_{n+1} . If z_{n+1} is nearly parallel to z_n , this axis may simply be permuted with other axes, such as y_{n+1} , to avoid using the Hayati parameters.

In addition to the position equations (14.12), orientation equations of coordinate system $n+1$

$${}^{-1}\mathbf{R}_{n+1}^l = \mathbf{F}(q^l, \boldsymbol{\phi}), \quad l = 1, \dots, P \quad (14.20)$$

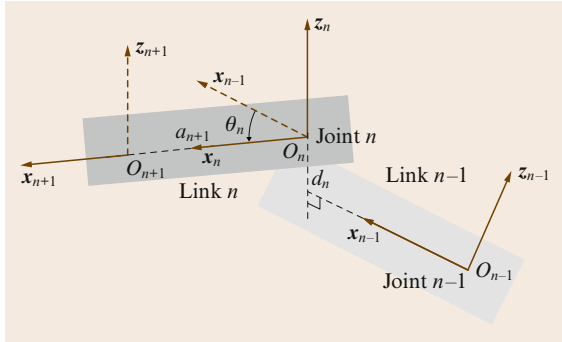


Fig. 14.4 A measured point O_{n+1} is located on the end link by adding a coordinate system $n+1$

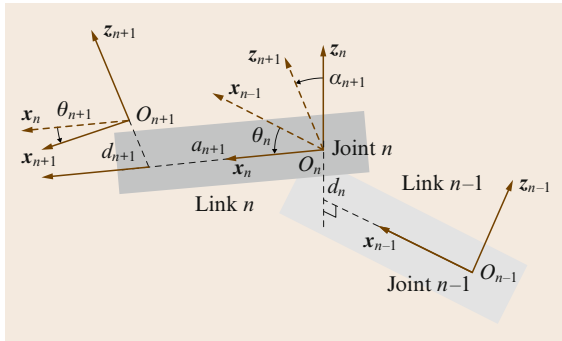


Fig. 14.5 Full pose measurement of coordinate system $n+1$ in the end link

are extracted from ${}^{-1}T_{n+1}$, where F is a matrix function. Linearization of this equation yields:

$$\begin{aligned}\Delta^{-1}R_{n+1}^l &= {}^{-1}R_{n+1}^l - {}^{-1}R_{n+1,c}^l \\ &= \Delta^{-1}\rho_{n+1}^l \times {}^{-1}R_{n+1,c}^l \\ S(\Delta^{-1}\rho_{n+1}^l) &= ({}^{-1}R_{n+1}^l - {}^{-1}R_{n+1,c}^l)({}^{-1}R_{n+1,c}^l)^\top, \end{aligned} \quad (14.21)$$

where $\Delta^{-1}\rho_{n+1}^l$ are differential orthogonal rotations which are the finite-difference counterpart to the angular velocity vector. Continuing with this analogy, a Jacobian J similar to that for spatial velocities (Sect. 1.8) is employed to represent the combined effect of parameter variations $\Delta\phi$ on changes in position $\Delta^{-1}p_{n+1}^l$ and orientation $\Delta^{-1}\rho_{n+1}^l$:

$$\begin{pmatrix} \Delta^{-1}p_{n+1}^l \\ \Delta^{-1}\rho_{n+1}^l \end{pmatrix} = J^l \Delta\phi. \quad (14.22)$$

Compared to (14.13), the Jacobian J^l now has six rows, as do the individual parameter Jacobians:

$$J_{a_i}^l = \begin{cases} \begin{pmatrix} {}^{-1}x_{i-1}^l \\ \mathbf{0} \end{pmatrix} & \text{DH} \\ \begin{pmatrix} {}^{-1}x_{i-1}^l \\ \mathbf{0} \end{pmatrix} & \text{Hayati} \end{cases}, \quad (14.23)$$

$$J_{d_i}^l = \begin{pmatrix} {}^{-1}z_i^l \\ \mathbf{0} \end{pmatrix}, \quad (14.24)$$

$$J_{\alpha_i}^l = \begin{cases} \begin{pmatrix} {}^{-1}x_{i-1}^l \times {}^{-1}d_{i-1,n+1}^l \\ {}^{-1}x_{i-1}^l \end{pmatrix} & \text{DH} \\ \begin{pmatrix} {}^{-1}x_{i-1}^l \times {}^{-1}d_{i-1,n+1}^l \\ {}^{-1}x_{i-1}^l \end{pmatrix} & \text{Hayati} \end{cases}, \quad (14.25)$$

$$J_{\theta_i}^l = \begin{pmatrix} {}^{-1}z_i^l \times {}^{-1}d_{i,n+1}^l \\ {}^{-1}z_i^l \end{pmatrix}, \quad (14.26)$$

$$J_{\beta_i}^l = \begin{pmatrix} {}^{-1}y_{i-1}^l \times {}^{-1}d_{i-1,n+1}^l \\ {}^{-1}y_{i-1}^l \end{pmatrix}. \quad (14.27)$$

As before, $\Delta\phi$ is found by least squares and ϕ by iteration.

Closed-Loop Calibration

Physical constraints on end-effector position or orientation can substitute for measurements. The location of

a physical constraint defines the reference coordinate system; the *measurement* of endpoint position or orientation is therefore defined as zero. The deviation from the physical constraint due to an incorrect kinematic model is cast as a displacement from the reference coordinates. Analogous to point measurement and full pose measurement, there are closed-loop methods using point constraints and full pose constraints.

Point Constraint. Suppose that the end-effector holds a stylus that makes contact with a fixed point in the environment. The orientation of the stylus can be changed by varying the joint angles, as long as the point of contact is not changed. Before, the measurement system for point measurement defined the reference coordinate system -1 (Fig. 14.3), and there was an end effector coordinate system $n+1$ whose origin O_{n+1} was measured (Fig. 14.4). Now the reference coordinate origin O_0 has index 0 and is collocated with O_{n+1} (Fig. 14.6). An extra coordinate system -1 is not required because a point is located relative to coordinate system 1 with only three parameters: a_1 , d_1 , and θ_1 . The arbitrary choice $\alpha_1 = 0$ is made, i. e., z_0 is made parallel to z_1 .

Different poses while maintaining the point contact can be generated manually, or they can be generated automatically if there is a force control capability. In comparison to (14.13), the *measured* position ${}^0p_{n+1}^l = 0$ by definition and the linearized calibration equation is simply:

$$\Delta^{-1}p_{n+1}^l = -{}^{-1}p_{n+1,c}^l = J^l \Delta\phi. \quad (14.28)$$

The set of poses that may be generated is limited relative to open-loop calibration, which may affect identifiability.

Full Pose Constraint. Analogous to full pose measurement (Fig. 14.5), the end link may be fully constrained

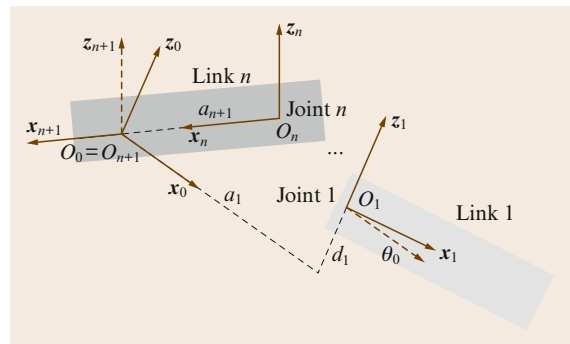


Fig. 14.6 Fixed point contact, setting $O_0 = O_{n+1}$

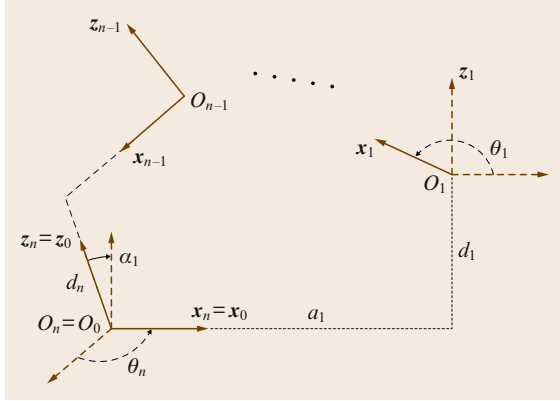


Fig. 14.7 Fully constrained end link, setting $O_0 = O_n$

by rigidly gripping the environment. If the manipulator is redundant (Chap. 11), then poses can be generated through self motion. Generating such poses would require endpoint force/torque sensing or joint torque sensing to be accomplished.

The end link can be considered part of the ground due to rigid attachment, and therefore one fewer coordinate system is required than for the fixed point case. Figure 14.7 shows one way of setting up coordinate systems 0 and n for calibration. Axis z_0 is set equal and coincident with axis z_n . The common normal between z_0 and z_1 sets the origin O_0 and the axis x_0 . Coordinate system n is completed by defining $O_n = O_0$ and $x_n = x_0$. Six parameters result for calibration, as they must to relate coordinate system n to coordinate system 0: θ_n , d_n , α_1 , a_1 , d_1 , and θ_1 .

Similar to (14.28) after index adjustment, the *measured* position ${}^0p_n^l = 0$ by definition and the linearized position calibration equation is:

$$\Delta {}^0p_n^l = -{}^0p_{n,c}^l. \quad (14.29)$$

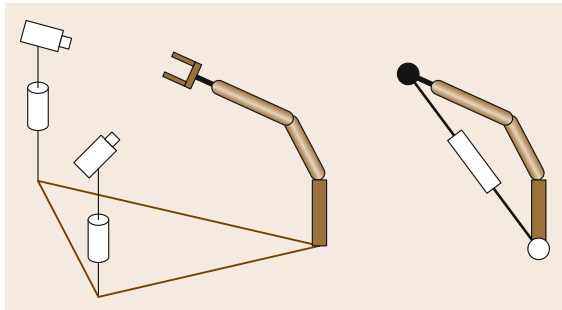


Fig. 14.8 External metrology system modeled as a 6-DOF joint

With regard to the error in orientation (14.21), the *measured* orientation ${}^0R_n^l = I$, the identity matrix, after index adjustment.

$$S(\Delta {}^0\rho_n^l) = (I - {}^0R_{n,c}^l)({}^0R_{n,c}^l)^\top = ({}^0R_{n,c}^l)^\top - I. \quad (14.30)$$

The error equation (14.22) is then applied as for full pose measurement, after index adjustment. The set of poses that may be generated is limited relative to open-loop calibration, which may affect identifiability.

14.2.2 Parallel Manipulator Calibration

Parallel manipulators are comprised of multiple closed loops. The methods of the previous section could readily be extended to calibrate parallel manipulators. Rather than treat parallel and serial manipulators differently, and continue to elaborate the calibration equations on a case-by-case basis that includes different loop arrangements and different sensing arrangements, *Hollerbach* and *Wampler* [14.8] presented a unifying methodology termed the *calibration index*, which views all calibration problems as closed-loop calibration problems.

Open-loop calibration is converted to closed-loop calibration by considering the end effector measurement to form a joint. All measurements, whether from joints or metrology systems, are put on an equal footing, as are all unsensed joints, whether from unsensed components of pose, passive environmental constraints, or joints in the chain without sensors. In the case of parallel linkages, sufficient numbers of loop-closure equations are formulated to characterize the kinematics and are combined at each pose. Because the loop-closure equations are implicit functions of all measurements, *Wampler* et al. [14.2] called this calibration method the *implicit loop method*.

Figure 14.8 illustrates this idea for a calibrated stereo camera metrology system that measures the 3-D coordinates of a distinguished point on the end-effector of an uncalibrated robot. On the right this camera system has been replaced by a prismatic leg, which stylistically represents a six-degree-of-freedom (6-DOF) joint that provides equivalent 3-D coordinate measurements. The result is a closed-loop mechanism.

The kinematic loop closure equations f for the i th pose ($i = 1, \dots, P$) are formed as

$$f^i(\phi) \equiv f(x^i, \phi) = 0, \quad (14.31)$$

where ϕ is a vector of robot parameters to be calibrated, x^i is a vector of joint sensor readings and possibly external sensor readings, and f^i absorbs the sensor readings

\mathbf{x}^i and so is given an index. Combining (14.31) for the P poses into a single matrix equation

$$\mathbf{f}(\boldsymbol{\phi}) = \begin{pmatrix} \mathbf{f}^1 \top & \dots & \mathbf{f}^P \top \end{pmatrix}^\top = \mathbf{0}. \quad (14.32)$$

Linearize (14.32) around the nominal values of the parameters

$$\Delta \mathbf{f} = \frac{\partial \mathbf{f}}{\partial \boldsymbol{\phi}} \Delta \boldsymbol{\phi} = \mathbf{A} \Delta \boldsymbol{\phi}, \quad (14.33)$$

where $\Delta \mathbf{f}$ is the deviation of the computed loop closure equations from zero, \mathbf{J} is the identification Jacobian, and $\Delta \boldsymbol{\phi}$ is the correction to be applied to the current parameter estimate. The calibration problem is then solved by minimizing $\Delta \mathbf{f}$ via iterative least squares.

The Calibration Index

The basis for kinematic calibration is to compute the error in positioning a manipulator using the current kinematic model. The error may be relative to measurements by an external metrology system (open-loop methods), or relative to a physical constraint such as contact with a plane (closed-loop methods). An external metrology system may measure all six components of pose or fewer components such as a point on the end link (three components). Likewise, a physical constraint can limit between one and six components of pose. Constraints and measurements may be mixed. The number of measurements and constraints determine how many scalar equations per pose are available for calibration.

The calibration index C quantifies the number of equations per pose. While this analysis is rather straightforward for serial linkages, for parallel linkages it can be difficult to infer the number of independent equations per pose

$$C = S - M, \quad (14.34)$$

where S is the sensor index and M is the mobility index. The mobility index [14.17] characterizes the degrees of freedom in the calibration setup.

$$M = 6n - \sum_{i=1}^{N_J} n_i^c, \quad (14.35)$$

where n is the number of links, N_J is the number of joints, and n_i^c is the number of constraints at joint i . n includes any extra links attached to the robot to constrain or measure its motion. N_J includes joints of any additional linkages added for calibration. For a rotational or prismatic joint $n_i^c = 5$, while for a ball or spherical joint $n_i^c = 3$. For an external measurement system for a freely

moving end-effector $n_{N_J}^c = 0$, while for rigid attachment of the endpoint $n_{N_J}^c = 6$. While generally correct, there are exceptions to (14.35) due to special or degenerate mechanisms that have to be analyzed on a case-by-case basis (Chap. 12).

The sensor index S is the total number of sensors in the joints:

$$S = \sum_{i=1}^{N_J} S_i, \quad (14.36)$$

where S_i is the number of sensed degrees in joint i . Usually $S_i = 1$ for the lower-order pairs typical of actuated joints, while $S_{N_J} = 6$ for full pose measurement of the end-effector joint N_J . For an unsensed joint, such as in passive environment kinematics, $S_i = 0$.

If P is the number of poses, then \mathbf{CP} is the total number of equations for the calibration procedure. Clearly a larger C means that fewer poses are required, other things being equal. For the single-loop case consisting of a series of sensed lower-order robot joints ($S_i = 1, n_i^c = 5, i = 1, \dots, N_J - 1$) with a final joint ($S_{N_J}, n_{N_J}^c$) connecting the end-effector to ground, one has from (14.34–14.35)

$$C = S_{N_J} + n_{N_J}^c. \quad (14.37)$$

According to the calibration index, using full endpoint constraints is an equivalent kinematic calibration method to using full pose measurements. There is potentially a problem with the smaller range of poses available with the endpoint constrained, but otherwise the mathematics are the same.

Categorization of Serial Link Calibration Methods

A great variety of calibration methods have been proposed, varying in the manner of pose measurement or endpoint constraints. These are categorized below according to the calibration index C and the values of $n_{N_J}^c$ and S_{N_J} .

$C = 6$: $n_{N_J}^c = 0$ and $S_{N_J} = 6$ corresponds to full pose measurement. $n_{N_J}^c = 6$ and $S_{N_J} = 0$ corresponds to rigid endpoint attachment.

$C = 5$: $n_{N_J}^c = 0$ and $S_{N_J} = 5$ corresponds to 5-DOF pose measurement [14.14]. $n_{N_J}^c = 5$ and $S_{N_J} = 0$ corresponds to 5-DOF endpoint constraints, such as manipulation of an unsensed passive hinge joint [14.18].

$C = 4$: No published method exists for $C = 4$.

$C = 3$: $n_{N_J}^c = 0$ and $S_{N_J} = 3$ corresponds to 3-DOF pose measurement. $n_{N_J}^c = 3$ and $S_{N_J} = 0$ corresponds to 3-DOF endpoint constraints.

- $C = 2$: $n_{N_j}^c = 0$ and $S_{N_j} = 2$ corresponds to 2-DOF pose measurements, such as is provided by a single theodolite [14.12]. $n_{N_j}^c = 2$ and $S_{N_j} = 0$ corresponds to 2-DOF endpoint constraints. Motion along a line provides two constraints [14.19].
- $C = 1$: $n_{N_j}^c = 0$ and $S_{N_j} = 1$ corresponds to measurement of just 1-DOF pose, provided by a linear transducer such as an linear variable differential transformer (LVDT) [14.9] or wire potentiometer [14.10]. $n_{N_j}^c = 1$ and $S_{N_j} = 0$ corresponds to a plane constraint [14.20].

Categorization of Parallel-Link Calibration Methods

To calibrate parallel robots, a loop-closure equation is written for each loop j :

$$\mathbf{0} = f_j(\phi). \quad (14.38)$$

The equations for all the loops are combined. One problem is to eliminate unsensed degrees of freedom. The calibration index is now applied based on the number of loops.

- 2 loops: A two-loop mechanism has three arms or legs, attached to a common platform. An example is the RSI Research Ltd. Hand Controller [14.21], which employs three 6-DOF arms with three sensed joints each. The mobility of this mechanism is $M = 6$. As $S = 9$, therefore $C = 3$ and closed-loop calibration is possible.
- 4 loops: Nahvi et al. [14.22] calibrated a spherical shoulder joint, driven redundantly by four

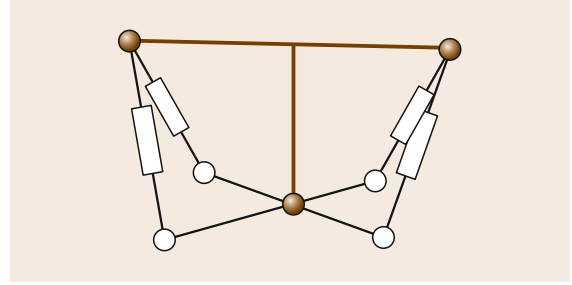


Fig. 14.9 Redundant parallel drive shoulder joint

prismatic legs (Fig. 14.9). In addition, the platform is constrained to rotate about a spherical joint; with the four legs, four kinematic loops are formed. For this system, $M = 3$ and $S = 4$, so that $C = 1$. Hence self-calibration is possible. Without the extra leg and the sensing it provides, one would have $C = 0$ and calibration would not be possible.

- 5 loops: Wampler et al. [14.2] calibrated the six-legged Stewart platform ($M = 6$) via a closed-loop procedure. In addition to leg length measurements, all angles on one of the legs were measured ($S = 11$). The reason for the extra sensors is to yield a unique forward kinematics solution, but a side benefit is that, with $C = 5$, closed-loop calibration is possible. For a regular Stewart platform without the instrumented leg, $S = 6$ and hence $C = 0$. External pose measurement is required; for example, with full pose measurement $S = 12$ and $C = 6$; full pose measurement was utilized in [14.23].

14.3 Inertial Parameter Estimation

A rigid body i has 10 inertial parameters: mass m_i , center of mass \mathbf{r}_{0i} relative to an origin O_0 , and symmetric inertia matrix \bar{I}_i referred here to the origin O_i (Fig. 14.11). The rigid body may be a load held by the end effector, or it can be one of the manipulator's own links. By generating a trajectory and measuring forces or torques in combination with velocity and acceleration, some or all of the inertial parameters can be estimated.

14.3.1 Load Inertial Parameter Estimation

Beginning with load inertial parameter estimation sets the stage for link inertial parameter estimation. It is as-

sumed that there is a wrist-mounted six-axis force/torque sensor, and that suitable filters have been designed to estimate the velocity $\dot{\theta}_i$ and acceleration $\ddot{\theta}_i$ at each joint i . The procedure involves

1. formulating the Newton–Euler equations of the load dynamics to reveal a linear dependence on the inertial parameters, and
2. estimating the parameters using ordinary least squares.

Load Dynamics

Kinematically, the force-torque sensor is part of the last link n , mounted near the joint axis z_n and origin O_n . The

sensor provides readings relative to its own coordinate system, called $n + 1$ (Fig. 14.10). The remainder of the last link is attached to the shaft of the force-torque sensor, and so the force-torque sensor is measuring the loads on the last link excluding itself. The acceleration of the force-torque sensor reference frame would have to be calculated based on the amount of offset from O_n , but we will ignore that complication here and assume that the sensor origin O_{n+1} is coincident with O_n .

The center of mass of link n is defined as C_n , located relative to the base origin O_0 by $r_n = C_n - O_0$ and relative to link n 's origin O_n by $c_n = C_n - O_n$. All vectors and matrices are expressed in coordinate system n , in which the center of mass location c_n and the inertia matrix \bar{I}_n about origin O_n are constant. The force-torque sensor measures the spatial force f_n at O_n in the coordinate system of the sensor, but the readings are presumed to be converted to the link n coordinates.

From (2.33) in Chap. 2, the Newton–Euler equations referred to O_n are

$$f_n = I_n a_n + v_n \times I_n v_n. \quad (14.39)$$

Substituting for the spatial inertia I_n , the spatial acceleration a_n , and the spatial velocity v_n , the first term on the right evaluates to:

$$\begin{aligned} I_n a_n &= \begin{pmatrix} \bar{I}_n & m_n S(c_n) \\ m_n S(c_n)^\top & m_n I \end{pmatrix} \begin{pmatrix} \dot{\omega}_n \\ \dot{v}_n \end{pmatrix} \\ &= \begin{pmatrix} \bar{I}_n \dot{\omega}_n + m_n S(c_n) (\ddot{d}_{0n} - \omega_n \times v_n) \\ m_n S(c_n)^\top \dot{\omega}_n + m_n (\ddot{d}_{0n} - \omega_n \times v_n) \end{pmatrix}, \end{aligned}$$

where $\dot{v}_n = \ddot{d}_{0n} - \omega_n \times v_n$ has been substituted. The second term on the right evaluates to:

$$\begin{aligned} v_n \times I_n v_n &= \begin{pmatrix} S(\omega_n) & S(v_n) \\ \mathbf{0} & S(\omega_n) \end{pmatrix} \\ &\quad \times \begin{pmatrix} \bar{I}_n & m_n S(c_n) \\ m_n S(c_n)^\top & m_n I \end{pmatrix} \begin{pmatrix} \omega_n \\ v_n \end{pmatrix} \\ &= \begin{pmatrix} S(\omega_n) \bar{I}_n \omega_n + m_n S(c_n) S(\omega_n) v_n \\ S(\omega_n) m_n S(c_n)^\top \omega_n + S(\omega_n) m_n v_n \end{pmatrix}. \end{aligned}$$

Combining and simplifying,

$$f_n = \begin{pmatrix} \bar{I}_n \dot{\omega}_n + S(\omega_n) \bar{I}_n \omega_n - S(\ddot{d}_{0n}) m_n c_n \\ m_n \ddot{d}_{0n} + S(\dot{\omega}_n) m_n c_n + S(\omega_n) S(\omega_n) m_n c_n \end{pmatrix}, \quad (14.40)$$

where the mass moment $m_n c_n$ appears as a quantity to be estimated in combination. However, since the mass m_n

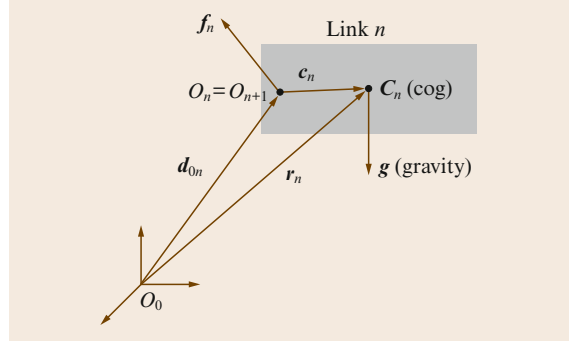


Fig. 14.10 Dynamics of the last link

is separately estimated from the $m_n \ddot{d}_{0n}$ term, the center of mass c_n can be extracted. To account explicitly for gravity g , we substitute $\ddot{d}_{0n} - g$ for \ddot{d}_{0n} subsequently.

To formulate an estimation algorithm, the force and torque measured by the wrist sensor must be expressed in terms of the product of known geometric parameters and the unknown inertial parameters. Elements of the inertia matrix are vectorized as $I(\bar{I}_n)$ according to the following notation:

$$\begin{aligned} \bar{I}_n \omega_n &= \begin{pmatrix} \omega_1 & \omega_2 & \omega_3 & 0 & 0 & 0 \\ 0 & \omega_1 & 0 & \omega_2 & \omega_3 & 0 \\ 0 & 0 & \omega_1 & 0 & \omega_2 & \omega_3 \end{pmatrix} \begin{pmatrix} I_{11} \\ I_{12} \\ I_{13} \\ I_{22} \\ I_{23} \\ I_{33} \end{pmatrix} \\ &\equiv L(\omega_n) I(\bar{I}_n), \end{aligned}$$

where $L(\omega_n)$ is a 3×6 matrix of angular velocity elements and

$$\bar{I}_n = \begin{pmatrix} I_{11} & I_{12} & I_{13} \\ I_{12} & I_{22} & I_{23} \\ I_{13} & I_{23} & I_{33} \end{pmatrix}.$$

Using these expressions, (14.40) can be written as:

$$\begin{aligned} f_n &= \begin{pmatrix} \mathbf{0} & -S(\ddot{d}_{0n}) & L(\dot{\omega}_n) + S(\omega_n) L(\omega_n) \\ \ddot{d}_{0n} & S(\dot{\omega}_n) + S(\omega_n) S(\omega_n) & \mathbf{0} \end{pmatrix} \\ &\quad \times \begin{pmatrix} m_n \\ m_n c_n \\ I(\bar{I}_n) \end{pmatrix} \end{aligned}$$

or more compactly,

$$f_n = A_n \phi_n, \quad (14.41)$$

where A_n is a 6×10 matrix, and ϕ_n is the vector of the 10 unknown inertial parameters which appear linearly.

Estimating the Parameters

The quantities inside the A_n matrix are computed by direct kinematics computation from the measured joint angles, and from the estimated joint velocities and accelerations. The estimation is typically done by bandpass filtering of the joint angle data [14.24]. The elements of the f_n vector are measured directly by the wrist force sensor. For robust estimates in the presence of noise, a larger number of data points are obtained by moving the manipulator through a suitable trajectory. Augment f_n and A_n as:

$$A = \begin{pmatrix} A_n^1 \\ \vdots \\ A_n^P \end{pmatrix}, \quad f = \begin{pmatrix} f_n^1 \\ \vdots \\ f_n^P \end{pmatrix}, \quad (14.42)$$

where P is the number of data points. The least-squares estimate for ϕ is given by

$$\hat{\phi} = (A^T A)^{-1} A^T f. \quad (14.43)$$

Equation (14.43) can also be formulated in a recursive form for online estimation.

For object recognition, it may be desired to derive the inertia about the center of mass, which can be done with the parallel axis theorem. An eigenvalue analysis can diagonalize the inertia matrix to reveal the principal axes and inertia.

14.3.2 Link Inertial Parameter Estimation

By treating each link i of a manipulator as a load, the previous formulation can be extended to the link estimation problem. Unlike load estimation, the only sensing is one component of joint torque. This lack of full force/torque sensing, along with the restricted movement near the base, makes it impossible to find all the inertial parameters of the proximal links. The missing parameters are unimportant, since they have no effect on the control of the arm.

For geared electric drives, joint torque may either be measured by joint torque sensors (Sect. 19.2.4) or estimated from the motor current by using an electric motor model (Sect. 6.3). Most robots do not have joint torque sensors, in which case joint friction needs to be taken into account. Joint friction typically consumes a large fraction of the torque that the motor produces. Coulomb and viscous friction are the most important components of a friction model, although Stribeck friction may need to

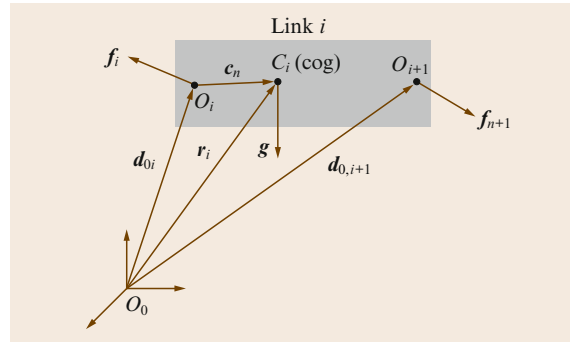


Fig. 14.11 Location of the center of gravity for an intermediate link i , and the constraint forces and torques

be modeled for low joint velocities [14.25,26]. A friction model is estimated through a process of moving one joint at a time, and relating motor torque to velocity. Ripple torque, due either to uncompensated inhomogeneities of the magnetic field in the motor [14.27,28], or to positional dependencies of gear tooth interaction [14.25], may need to be modeled.

Consider a manipulator with n joints (Fig. 14.11). (Only manipulators with revolute joints will be considered, since handling prismatic joints requires only trivial modifications to the algorithm.) Define f_{ij} as the spatial force at joint i due to movement of link j alone. Then f_{ii} is the spatial force at joint i due to movement of its own link, and is the same as (14.41) with i substituted for n in the A_n matrix

$${}^i f_{ii} = {}^i A_i \phi_i, \quad (14.44)$$

where ϕ_i is the vector of unknown link i inertial parameters. Superscript i has been added to indicate that vectors are expressed in terms of link i coordinates, so that the center of mass ${}^i c_i$ and the inertia matrix ${}^i \bar{I}_i$ are constant.

The total spatial force ${}^i f_i$ at joint i is the sum of the spatial forces ${}^i f_{ij}$ for all links j distal to joint i :

$${}^i f_i = \sum_{j=i}^n {}^i f_{ij}. \quad (14.45)$$

Each spatial force ${}^i f_{ij}$ at joint i is determined by transmitting the distal spatial force ${}^j f_{jj}$ across intermediate joints. Using the spatial force transform matrix ${}^i X_j^F$,

$${}^i f_{i,i+1} = {}^i X_{i+1}^F {}^{i+1} f_{i+1,i+1} = {}^i X_{i+1}^F {}^{i+1} A_{i+1} \phi_{i+1}. \quad (14.46)$$

For convenience, we note that ${}^i X_i^F = I_{6 \times 6}$. To obtain the forces and torques at the i -th joint due to the movements

of the j -th link, these matrices can be cascaded:

$$\begin{aligned} {}^i f_{ij} &= {}^i X_{i+1}^F \quad {}^{i+1} X_{i+2}^F \quad \dots \quad {}^{j-1} X_j^F \quad {}^j f_{jj} \\ &= {}^i X_j^F A_j \phi_j. \end{aligned} \quad (14.47)$$

An upper-diagonal matrix expression for a serial kinematic chain can be derived from (14.45) and (14.47):

$$\begin{pmatrix} {}^1 f_1 \\ {}^2 f_2 \\ \vdots \\ {}^n f_n \end{pmatrix} = \begin{pmatrix} {}^1 X_1^F A_1 & {}^1 X_2^F A_2 & \dots & {}^1 X_n^F A_n \\ \mathbf{0} & {}^2 X_2^F A_2 & \dots & {}^2 X_n^F A_n \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & {}^n X_n^F A_n \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_n \end{pmatrix}. \quad (14.48)$$

This equation is linear in the unknown parameters, but the left-hand side is composed of a full force-torque vector at each joint. Since only the torque τ_i about the joint rotation axis z_i can usually be measured, each spatial force ${}^i f_i$ must be projected onto the joint rotation axis, reducing (14.48) to

$$\tau = K\phi, \quad (14.49)$$

where

$$\begin{aligned} \tau_i &= \begin{pmatrix} z_i \\ \mathbf{0} \end{pmatrix} \cdot f_i, \\ K_{ij} &= \begin{pmatrix} z_i \\ \mathbf{0} \end{pmatrix} \cdot {}^i X_j^F A_j, \quad \phi = \begin{pmatrix} \phi_1 \\ \vdots \\ \phi_n \end{pmatrix} \end{aligned}$$

and $K_{ij} = \mathbf{0}_{1 \times 10}$ if $i > j$. For an n -link manipulator, τ is an $n \times 1$ vector, ϕ is a $10n \times 1$ vector, and K is an $n \times 10n$ matrix.

Equation (14.49) represents the dynamics of the manipulator for one sample point. As with load identification, (14.49) is augmented using P data points:

$$K = \begin{pmatrix} K^1 \\ \vdots \\ K^P \end{pmatrix}, \quad \tau = \begin{pmatrix} \tau^1 \\ \vdots \\ \tau^P \end{pmatrix},$$

where τ is now an $nP \times 1$ vector and K is now $nP \times 10n$.

Unfortunately, one cannot apply simple least-squares estimation because $K^T K$ is not invertible due to the loss of rank from restricted degrees of freedom at the

proximal links and the lack of full force-torque sensing. Some inertial parameters are completely unidentifiable, while some others can only be identified in linear combinations. The general topic of identifiability of parameters, and how to handle unidentifiable parameters, is discussed next.

An issue with geared electric drives is the rotor inertia. If not known, the rotor inertia can be added to the list of 10 inertial parameters to be identified for a link [14.29]. For large gear ratios, the rotor inertia can dominate the other components of link inertia.

14.3.3 Link Parameter Estimation for More Complex Structures

In this section we present the dynamic identification model for kinematic tree (spanning tree) robots and kinematic closed-loop robots including parallel robots. These models are linear in the inertial parameters and can be represented by a model similar to (14.49), thus these parameters can be identified using the same techniques.

Tree-Structured Robots

For an n -link manipulator with a kinematic tree structure, the dynamic identification model equations (14.48) must be modified to take into account that the inertial parameters of link j have no effect on the dynamics of links which do not belong to the chain from the base to that link. For such structures, the link on which link j is articulated is denoted by $p(j)$ (parent of j – see Chap. 2). It could be a link with any number i , such that $i < j$. Thus the nonzero elements of the column of the matrix representing the coefficients of the inertial parameters of link j in (14.48) would be:

$${}^j X_j^F A_j, {}^{p(j)} X_j^F A_j, {}^{p(p(j))} X_j^F A_j, \dots, {}^b X_j^F A_j, \quad (14.50)$$

where b is the first link of the chain connecting link 0 to link j , thus $p(b) = 0$.

Consequently, in (14.49) of a tree robot we obtain: $K_{ij} = \mathbf{0}_{1 \times 10}$ if $i > j$ or if link i does not belong to the branch connecting link 0 to link j . This means that some elements of the upper-right submatrix of K will be zero.

We note that a serial structure is a particular case of a tree structure where $p(j) = j - 1$. Thus any link m such that $m < j$ will belong to the chain from the base to link j .

Closed-Loop Robots

The dynamic model of a closed-loop structure can be obtained by considering first an equivalent spanning tree

by opening one joint at each closed loop, and then using the principle of virtual work such that:

$$\tau = G^\top K_{tr} \phi, \quad G = \left(\frac{\partial q_{tr}}{\partial q_a} \right) \quad (14.51)$$

with q_a an $N \times 1$ vector of N active joint angles (N differs from n , the total number of joints), and q_{tr} an $n \times 1$ vector of the joints of the spanning tree structure.

Parallel Robots

A parallel robot is a particular case of closed-loop robots (Chap. 12). It is composed of a moving platform, representing the terminal link, that is connected to the base by m parallel legs. The dynamic model of parallel robots can be given by [14.30]

$$\tau = J^\top A_p \phi_p + \sum_{i=1}^m \left(\frac{\partial q_i}{\partial q_a} \right)^\top K_i \phi_i, \quad (14.52)$$

where $K_i \phi_i$ represents the dynamic model of leg i with K_i a function of $(q_i, \dot{q}_i, \ddot{q}_i)$, q_i is the vector of joint angles of leg i , $A_p \phi_p$ is the Newton–Euler spatial force of the platform calculated in terms of the Cartesian variables of the mobile platform using (14.41), and J is the Jacobian matrix of the parallel manipulator. We can obtain (14.52) from (14.51) by supposing that the span-

ning tree structure is obtained by separating the mobile platform from the legs.

Equation (14.52) can be rewritten as

$$\tau = \left(J^\top A_p \quad (\partial q_1 / \partial q_a)^\top K_1 \quad \cdots \quad (\partial q_m / \partial q_a)^\top K_m \right) \phi_{par} \quad (14.53)$$

$$\tau = K_{par}(\omega_p, \dot{v}_p, \dot{\omega}_p, q_i, \dot{q}_i, \ddot{q}_i) \phi_{par},$$

where ϕ_{par} is the vector of the inertial parameters of the robot (the legs and the platform).

$$\phi_{par} = \begin{pmatrix} \phi_p \\ \phi_1 \\ \vdots \\ \phi_m \end{pmatrix}.$$

In the common case where the legs are identical and their inertial parameters are denoted by ϕ_{leg} , the identification model can be rewritten by the following equation, which considerably reduces the number of inertial parameters to be identified:

$$\tau = \left(J^\top A_p \quad \sum_{i=1}^m (\partial q_i / \partial q_a)^\top K_i \right) \begin{pmatrix} \phi_p \\ \phi_{leg} \end{pmatrix}. \quad (14.54)$$

The identification of the orthoglide parallel robot is given in [14.31].

14.4 Identifiability and Numerical Conditioning

That some inertial parameters in (14.49) are unidentifiable does not mean that they are intrinsically identifiable, only that the experimental setup does not allow them to be identified. Limited motion near the base could be fixed by placing the whole manipulator on a six-axis moving platform such as a Stewart–Gough platform. In fact, for mobile manipulators mounted on high-mobility vehicles such as satellites, it may be necessary to know the full inertial model of the manipulator. Additional sensors could be added, for example a six-axis force-torque sensor to the base of the robot [14.5], to identify some additional (but not all) inertial parameters.

A similar situation may arise in kinematic calibration, where for example joint models might be augmented to include gear eccentricities, transmission and coupling coefficients of gears, joint elasticity, link elasticity, and base deflection [14.32]. By instrumenting the robot with additional sensors, for example, by placing joint angle sensors before and after the gears

to measure joint deflection, it may be possible to identify additional parameters of interest. Additional sensing with a theodolite was done to measure base deflection in [14.12].

Given that the experimental setup is what it is, there may be no recourse but to deal with unidentifiable parameters, treated in the next section. Another problem is that parameters might in principle be identifiable, but numerical conditioning prevents their accurate determination. Issues such as adequacy of the collected data, and the effect of different units and magnitudes of parameters, are treated subsequently.

14.4.1 Identifiability

There are two main approaches towards handling unidentifiable parameters, depending on whether the goal is a structural model or a prediction model. For a structural model, the goal is to find the minimum

parameter set that provides a meaningful physical description of the system by eliminating parameters until all are identifiable. This is done by careful evaluation of the effect of each parameter of the original model.

For a prediction model, the goal is to match the outputs to the inputs as closely as possible, and is more of a curve-fitting approach. The resulting parameter values are not necessarily physically meaningful or accurate.

Structural Model

It may be possible in the initial model formation to avoid including redundant or unidentifiable parameters. At other times, it is not possible to determine a priori what the minimal parameter set is, because of system complexity or because of numerical difficulties such as measurement error or limitations in the collected data.

A Priori Parameter Elimination. In the initial model formation, the choice of representation can immediately dictate whether the model is redundant or minimal. For kinematic calibration, the minimal parameter set includes four parameters for a rotary joint and two parameters for a prismatic joint. The detailed presentation in Sect. 14.2.1 of how to set up the DH/Hayati parameters for different kinematic and sensing arrangements has the advantage that the parameter set is minimal (excepting joint models). Once done, the application to any manipulator is somewhat formulaic.

Five- and six-parameter joint models have also been proposed, either for the convenience of locating link coordinate systems or for the ease of model formation [14.13, 29, 33]. With such redundant parameter sets, extra steps must be taken to handle the numerical difficulties caused by the redundancy, such as by reducing the number of parameters in a postprocessing step. When there are complicated joint models that include gearing effects as mentioned above, the resultant large numbers of parameters make the parameter elimination problem more difficult. Determining the minimal set of inertial parameters offers a different sort of complexity, because large numbers of parameters are not identifiable or are identifiable only in linear combinations. Again, model reduction is one approach to handle this problem.

Two approaches towards determining a minimal or base parameter set are

1. numerical identification of the unidentifiable parameters or linear combinations of parameters
2. symbolic determination

Numerical identification involves rank reduction through matrix manipulation of the regressor, either

through QR decomposition or singular value decomposition [14.4]. If the kinematic or dynamic model is known exactly, then a simulation using exactly generated *data* will yield a noise-free regressor matrix \mathbf{A} in (14.3). For QR decomposition, the regressor matrix is factored as:

$$\mathbf{A} = \mathbf{Q} \begin{pmatrix} \mathbf{R} \\ \mathbf{0}_{MP-N_{\text{par}}, N_{\text{par}}} \end{pmatrix}, \quad (14.55)$$

where \mathbf{Q} is an $M \times MP$ orthogonal matrix, \mathbf{R} is an $N_{\text{par}} \times N_{\text{par}}$ upper-triangular matrix, and $\mathbf{0}_{MP-N_{\text{par}}, N_{\text{par}}}$ is the zero matrix with dimensions $MP - N_{\text{par}} \times N_{\text{par}}$. Theoretically, the unidentifiable parameters ϕ_i are those whose corresponding diagonal elements R_{ii} of matrix \mathbf{R} are zero. In practice, $|R_{ii}|$ is considered to be zero if it is less than the numerical zero ζ :

$$\zeta = MP\epsilon \max_i |R_{ii}|, \quad (14.56)$$

where ϵ is the computer precision [14.34]. Other parameters may appear in linear combinations depending on how many nonzero elements there are in row j of \mathbf{R} . Resolving these linear combinations is arbitrary. One approach is to zero all elements in the linear combination save one. The result is a prediction model rather than a structural model.

Symbolic determination of the base inertial parameters has been proposed in [14.35, 36]. Using an energy formulation, the total energy of link j is expressed as $\mathbf{h}_j \phi_j$, where \mathbf{h}_j is a row vector termed the total energy function whose elements are kinematic expressions of the angular and linear velocity of link j plus gravity. A recursive relation between neighboring links is developed as

$$\mathbf{h}_j = \mathbf{h}_{j-1}^{j-1} \lambda_j + \dot{q}_j \boldsymbol{\eta}_j, \quad (14.57)$$

where the elements of the 10×10 matrix $\mathbf{h}_{j-1}^{j-1} \lambda_j$ are functions of the DH parameters defining frame j , and the elements of the 1×10 row vector $\boldsymbol{\eta}_j$ depend on the linear and angular velocities. The details of these expressions can be found in [14.35, 36]. Grouping rules are then developed to find the precise linear combinations of parameters.

The minimum inertial parameters for a tree structure can be obtained using a closed-form solution similar to the serial structure case [14.36].

The term $\mathbf{K}_{\text{tr}} \phi$ in (14.51) shows that the minimum parameters of the spanning tree structure can be used to calculate the dynamics of the closed structure, giving by this a first reduction and grouping of the inertial parameters. However extra parameters can be eliminated

or grouped when considering the matrix \mathbf{G} . The case of parallelogram loops can be treated symbolically using closed-form relations [14.36]. For general closed loops, the minimum parameters must be determined using numerical methods such as QR decomposition.

In the case of parallel robots, we deduce from (14.52) that the minimum parameters of the legs can be used to calculate $\mathbf{K}_i \phi_i$, giving a first reduction of the inertial parameters. However, some other parameters can be grouped with the parameters of the platform. The minimum parameters of the Gough–Stewart robot are given in [14.37].

Data-Driven Parameter Elimination. The singular value decomposition of the regressor matrix can show which parameters are unidentifiable, weakly identifiable, or identifiable only in linear combination. For N_{par} parameters, P data points, and M -dimensional output measurements at each data point, the regressor matrix \mathbf{A} (14.3) or (14.7) can be decomposed as

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T, \quad (14.58)$$

where \mathbf{U} is an $MP \times MP$ orthogonal matrix, \mathbf{V} is an $N_{\text{par}} \times N_{\text{par}}$ orthogonal matrix, and $\mathbf{\Sigma}$ is the $MP \times N_{\text{par}}$ matrix of singular values

$$\mathbf{\Sigma} = \begin{pmatrix} \mathbf{S} \\ \mathbf{0}_{MP-N_{\text{par}}, N_{\text{par}}} \end{pmatrix}, \quad (14.59)$$

where $\mathbf{S} = \text{diag}(\mu_1, \dots, \mu_r, 0, \dots, 0)$ is the $N_{\text{par}} \times N_{\text{par}}$ matrix of ordered singular values, with μ_1 the largest and μ_r the smallest nonzero singular value. There may be $N_{\text{par}} - r$ zero singular values $\mu_{r+1} = \dots = \mu_{N_{\text{par}}} = 0$.

Especially when complex joint models are assumed that include flexibility, backlash, and gear eccentricity, it is not clear that all parameters can be identified. Retaining poorly identifiable parameters will degrade the robustness of the calibration; such parameters are indicated by zero or very small singular values. The expansion of (14.7) in terms of (14.58) is:

$$\Delta \mathbf{y} = \sum_{j=1}^r \mu_j (\mathbf{v}_j^T \Delta \phi) \mathbf{u}_j, \quad (14.60)$$

where \mathbf{u}_j and \mathbf{v}_j are the j -th columns of \mathbf{U} and \mathbf{V} . For zero or small singular values μ_j , the projection $\mathbf{v}_j^T \Delta \phi$ of the parameters onto the column vector \mathbf{v}_j in general represents a linear combination of parameters. It is also possible that only a single parameter results from the projection.

To proceed, it is first necessary to scale the parameters and the output measurements so that the

singular values are comparable. Scaling is discussed in Sect. 14.4.3. Small singular values signal that there are parameters that are poorly identifiable and that should be eliminated. By small singular values, *Schröer* [14.32] suggests the heuristic that the *condition number* of a well-conditioned regressor matrix should be less than 100:

$$\kappa(\mathbf{A}) = \frac{\mu_1}{\mu_r} < 100. \quad (14.61)$$

This heuristic derives from experience of their statistical community. If the condition number is above 100, the singular values are examined, beginning with the smallest one, which may be a zero singular value.

If the condition number is above 100, the linear sums (14.60) corresponding to the smallest singular value μ_r are examined. The elements of column \mathbf{v}_r are in one-to-one correspondence with the elements of $\Delta \phi$. If there is an element j of \mathbf{v}_r that is much larger than the others, then the parameter ϕ_j corresponding to that column element is a candidate for elimination. This procedure will tend to pinpoint parameters that are totally unidentifiable. Isolating the largest element of \mathbf{v}_r is once again only meaningful if the parameters have been previously scaled.

Once the parameter is eliminated, the condition number of the reduced regressor is computed again. The procedure iterates until the condition number of the regressor is below 100.

The previous procedure can be carried out using QR decomposition as in (14.55), by replacing ϵ representing the computer precision in (14.56) by a greater value function of the noise level.

Prediction Model

If there are several largest elements of \mathbf{v}_r that are roughly comparable in magnitude, then these parameters may only be identifiable in linear combination. There will be the same number of singular values which are too small. By examining several columns \mathbf{v}_j corresponding to the small singular values, these linear combinations can become apparent. The linear combinations can be resolved arbitrarily, i.e., all parameter elements in the linear combination save one can be set to zero. The result of zeroing some parameters is that the model is no longer a structural model, but rather a prediction model.

One may also proceed without first removing parameters. It can be shown by substituting the singular value decomposition that

$$(\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T = \mathbf{V} \left(\mathbf{S}^{-1} \mathbf{0}_{N_{\text{par}}, MP-N_{\text{par}}} \right) \mathbf{U}^T. \quad (14.62)$$

Consequently the solution to (14.8) can be expressed as [14.38]

$$\Delta\phi = \sum_{j=1}^{N_{\text{par}}} \frac{\mathbf{u}_j^\top \Delta\mathbf{y}}{\mu_j} \mathbf{v}_j. \quad (14.63)$$

One can see that poorly identifiable parameters corresponding to small singular values μ_j greatly perturb the estimates, because the weighting is $1/\mu_j$. The strategy is to remove their influence. If μ_j is zero or very small relative to the largest singular value μ_1 , then set $1/\mu_j = 0$.

Parameters that cannot be identified well are simply ignored in this procedure, which automatically converges to an identifiable parameter set. The resulting parameters can then be used for the model. A disadvantage is that the resulting parameters may not correspond to real model parameters.

Incorporating A Priori Parameter Estimates

Least squares treats parameter values as completely unknown, i. e., they could be anywhere in the range from $-\infty$ to $+\infty$. Yet often a fairly good initial estimate of the parameters is available, for example, from a manufacturer's specifications or in the case of a recalibration. It makes sense to incorporate such an a priori preference into the least-squares optimization [14.39].

Suppose there is a preference for a solution near $\phi = \phi_0$. Express the preference as $\mathbf{I}\phi = \phi_0$, where \mathbf{I} is the identity matrix, and append it as additional rows to (14.3) that reflect this preference

$$\begin{pmatrix} \mathbf{A} \\ \mathbf{I} \end{pmatrix} \phi = \begin{pmatrix} \mathbf{y} \\ \phi_0 \end{pmatrix}. \quad (14.64)$$

To proceed with the solution, we can treat ϕ_0 as a constant. Redefine the parameter vector as $\tilde{\phi} = \phi - \phi_0$, which we expect to be close to zero. Then

$$\begin{pmatrix} \mathbf{A} \\ \mathbf{I} \end{pmatrix} \tilde{\phi} = \begin{pmatrix} \tilde{\mathbf{y}} \\ \mathbf{0} \end{pmatrix}, \quad (14.65)$$

where $\tilde{\mathbf{y}} = \mathbf{y} - \mathbf{A}\phi_0$. We may not know ϕ_0 perfectly well, so we add a weighting parameter λ to express the confidence in this value

$$\begin{pmatrix} \mathbf{A} \\ \lambda \mathbf{I} \end{pmatrix} \tilde{\phi} = \begin{pmatrix} \tilde{\mathbf{y}} \\ \mathbf{0} \end{pmatrix}. \quad (14.66)$$

The larger is λ , the more confidence we have in our a priori estimate. The least-squares solution is therefore

$$\begin{aligned} \tilde{\phi} &= \left(\begin{pmatrix} \mathbf{A}^\top & \lambda \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{A} \\ \lambda \mathbf{I} \end{pmatrix} \right)^{-1} \begin{pmatrix} \mathbf{A}^\top & \lambda \mathbf{I} \end{pmatrix}^\top \begin{pmatrix} \tilde{\mathbf{y}} \\ \mathbf{0} \end{pmatrix} \\ &= (\mathbf{A}^\top \mathbf{A} + \lambda^2 \mathbf{I})^{-1} \mathbf{A}^\top \tilde{\mathbf{y}}. \end{aligned} \quad (14.67)$$

This solution is called *damped least squares*, where λ is the damping factor. Expanding this solution in terms of the singular value decomposition,

$$\tilde{\phi} = \sum_{j=1}^{N_{\text{par}}} (\mathbf{u}_j^\top \tilde{\mathbf{y}}) \frac{\mu_j}{\mu_j^2 + \lambda^2} \mathbf{v}_j. \quad (14.68)$$

Hence a very small μ_j is counteracted by the larger λ value; then the a priori information about parameter values dominates the information from the data, i. e., the data are ignored. Thus for damped least squares, no explicit action on the singular values is required, because the damping factor modifies the singular values. The solution will be perturbed over the normal least-squares solution by the magnitude of the λ choice.

Confidence Measure of Parameter Estimates

After calibration, an estimate $\hat{\mathbf{M}}$ of the covariance of the parameter estimates can be derived from the data [14.3]. Assume that the task variables $\Delta\mathbf{y}$ have previously been scaled for equal uncertainty, that there is no bias, and that the errors are uncorrelated. Then

$$\hat{\mathbf{M}} = \sigma^2 (\mathbf{A}^\top \mathbf{A})^{-1}. \quad (14.69)$$

An estimate of the standard deviation σ after running the calibration procedure is obtained from the χ^2 statistic [14.38, 40]:

$$\chi^2 = (\Delta\mathbf{y} - \mathbf{A}\Delta\hat{\phi})^\top (\Delta\mathbf{y} - \mathbf{A}\Delta\hat{\phi}). \quad (14.70)$$

An unbiased estimator for σ^2 is $\hat{\sigma}^2 = \chi^2/\nu$, where $\nu = MP - N_{\text{par}}$ is called the statistical *degrees of freedom*; ν subtracts from the total number of measurements MP the number N_{par} of parameters estimated, since some of the measurements went into determining ϕ .

The estimate $\hat{\mathbf{M}}$ can be used as a basis for eliminating parameters, by choosing those with the largest variances.

14.4.2 Observability

The measurements that are taken will influence the accuracy of parameter estimation. In kinematic calibration, the quality of the pose set has been measured by an *observability index*. In inertial parameter estimation, the

quality of the identification trajectory has been termed *persistent excitation* [14.41]. Regardless of whether data is collected statically as in kinematic calibration or dynamically as in inertial parameter estimation, the result is just a bunch of numbers that go into the regressor matrix, and so common terminology is appropriate.

In statistics, optimal experimental design theory has given rise to several data measures termed *alphabet optimalities* [14.42]. Some of the most prominent are the following.

- A-optimality minimizes the trace of $(A^T A)^{-1}$ to obtain regression designs.
- D-optimality maximizes the determinant of $(A^T A)^{-1}$.
- E-optimality maximizes the minimum singular value of $(A^T A)^{-1}$.
- G-optimality minimizes the maximum prediction variance, and does not have a simple expression in terms of singular values.

Although ties to the experimental design literature have not typically been made [14.43], several of the proposed observability indexes for robot calibration have an alphabet-optimality counterpart. A-optimality does not have a counterpart in the robot calibration literature, and conversely a few proposed observability indexes do not have an alphabet-optimality counterpart. In [14.43], it is proved that E-optimality and G-optimality are equivalent for exact design.

Borm and Menq [14.44, 45] proposed an observability index (here termed O_1 and numbered consecutively below) that maximizes the product of all of the singular values

$$O_1 = \frac{\sqrt{\mu_1 \cdots \mu_r}}{\sqrt{P}}. \quad (14.71)$$

This is similar to D-optimality. The rationale is that O_1 represents the volume of a hyperellipsoid in $\Delta \mathbf{y}$, defined by (14.7) when $\Delta \boldsymbol{\phi}$ defines a hypersphere, and the singular values represent the lengths of axes. Therefore maximizing O_1 gives the largest hyperellipsoid volume, and hence a good aggregate increase of the singular values. One can also derive O_1 from the well-known relation $\sqrt{\det(A^T A)} = \mu_1 \cdots \mu_r$.

Minimizing the condition number of A as a measure of observability has been proposed in [14.35, 46, 47]:

$$O_2 = \frac{\mu_1}{\mu_r}. \quad (14.72)$$

O_2 measures the eccentricity of the hyperellipsoid rather than its size. The intermediate singular values are not

considered to be that pertinent, because minimizing the condition number automatically makes all singular values become more similar in magnitude and makes the hyperellipsoid closer to a hypersphere.

Nahvi et al. [14.22] argue for maximizing the minimum singular value μ_r as an observability measure:

$$O_3 = \mu_r. \quad (14.73)$$

This is similar to E-optimality. The rationale is to make the shortest axis as long as possible, regardless of the other axes, that is to say, to improve the worst case. Consider the following standard result [14.4]

$$\mu_r \leq \frac{\|\Delta \mathbf{y}\|}{\|\Delta \boldsymbol{\phi}\|} \leq \mu_1 \quad (14.74)$$

or more particularly

$$\mu_r \|\Delta \boldsymbol{\phi}\| \leq \|\Delta \mathbf{y}\|. \quad (14.75)$$

Then maximizing μ_r ensures that a given size of parameter errors $\|\Delta \boldsymbol{\phi}\|$ has the largest possible influence on the pose errors $\|\Delta \mathbf{y}\|$.

Nahvi and Hollerbach [14.48] proposed the noise amplification index O_4 , which can be viewed as combining the condition number O_2 with the minimum singular value O_3 :

$$O_4 = \frac{\sigma_r^2}{\sigma_1}. \quad (14.76)$$

The rationale is to measure both the eccentricity of the ellipse through O_2 as well as the size of the ellipse through O_3 . The noise amplification index was argued to be the most sensitive to measurement error and modeling error.

Hollerbach and Lokhorst [14.21] found experimentally that the condition number and the minimum singular value gave about the same good result: their relative magnitudes were almost proportional to the root-mean-square (RMS) errors of the final parameter errors. The observability index O_1 was not as sensitive and not directly related to parameter errors. [14.43] derived general relations among the observability indexes and alphabet optimalities

$$O_1 \geq \text{A-optimality} \geq O_3. \quad (14.77)$$

They further showed that if $\mu_1 \geq 1$, then $O_3 \geq O_2$, and if $\mu_r \leq 1$, then $O_2 \geq O_4$. They also argued that O_3 (D-optimality) is in general the best index, because it minimizes the variance of the parameters and also minimizes the uncertainty of the end-effector pose.

Optimal Experimental Design

An observability index is typically used to decide how many data points to collect. As one begins to add data points, the observability increases, then plateaus. Adding additional data does not then improve the quality of the estimates. For kinematic calibration, data may be added through random selection, or optimal design methods can be employed that can drastically reduce the amount of data required [14.49, 50]. Optimal experimental designs work by measuring the effect of adding or exchanging data points [14.51].

Exciting Trajectories

For inertial parameter estimation, data points are not independent because they result from a trajectory rather than isolated poses. Therefore the issue is what type of trajectory to generate. Industrial robots usually have joint position point-to-point trajectory generators. A continuous and smooth trajectory is calculated by interpolating between these points, assuming zero initial and final velocity and acceleration at each point, and using polynomial interpolators. Excitation trajectories are obtained by minimizing an observability index, using nonlinear optimization techniques to calculate the polynomial coefficients, under the constraints of the joint positions, velocities, and accelerations limits [14.46].

It is possible to facilitate the optimization by proceeding to a sequential excitation procedure. Specific trajectories, which structurally excite a small number of parameters, are easier to optimize. For example, moving one joint at a time with constant velocities excites friction and gravity parameters. In this approach, sequential identification is avoided. However, it is better to collect all the data and to proceed to a globally weighted least-squares estimation [14.24]. This procedure avoids the accumulation of estimation errors and allows the calculation of the confidence intervals [14.69].

Specific trajectories have been proposed like sinusoidal interpolators [14.52], or periodic trajectories obtained from a spectral analysis of the contribution function of the parameters [14.53]. This is a general planning trajectory strategy, which is very important to get accurate experimental identification [14.54].

14.4.3 Scaling

The numerical conditioning of parameter estimation can be improved by scaling both the output measurements (task variable scaling) and the parameters.

Task Variable Scaling

When performing a least-squares analysis on the end-point pose error, position errors and orientation errors have to be combined (14.22):

$$\|\Delta \mathbf{y}^i\|^2 = \|\Delta^{-1} \mathbf{p}_{n+1}^l\|^2 + \|\Delta^{-1} \boldsymbol{\rho}_{n+1}^l\|^2. \quad (14.78)$$

However, position error and orientation error have different units, and so are not comparable. Furthermore, not all position or orientation components may be measured with equal accuracy.

Ordinary least squares (14.8) weighs all task variables equally. To weigh these variables differently, the general solution is left multiplication of (14.7) by a scaling matrix \mathbf{G} [14.39],

$$\begin{aligned} \mathbf{G} \Delta \mathbf{y} &= \mathbf{G} \mathbf{A} \Delta \boldsymbol{\phi}, \\ \Delta \tilde{\mathbf{y}} &\equiv \tilde{\mathbf{A}} \Delta \boldsymbol{\phi}, \end{aligned} \quad (14.79)$$

where $\Delta \tilde{\mathbf{y}} = \mathbf{G} \Delta \mathbf{y}$ is the scaled output vector and $\tilde{\mathbf{A}} = \mathbf{G} \mathbf{A}$ is the scaled regressor matrix. The weighted least-squares solution is

$$\Delta \boldsymbol{\phi} = (\tilde{\mathbf{A}}^\top \tilde{\mathbf{A}})^{-1} \tilde{\mathbf{A}}^\top \Delta \tilde{\mathbf{y}} = (\mathbf{A}^\top \mathbf{W} \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{W} \Delta \mathbf{y}, \quad (14.80)$$

where $\mathbf{W} = \mathbf{G}^\top \mathbf{G}$. Often \mathbf{W} is a diagonal matrix.

One approach to scaling position error versus orientation error is to equalize the effect of a parameter error $\Delta \phi_i$ on either position error or orientation error. Curiously, for human-sized robot arms the effect is equal without scaling due to an accident of metric units. If θ is the joint angle resolution, then $s = r\theta$ is the resulting endpoint position resolution. For human-sized arms, $r = 1$ m and hence $s = \theta$. Thus meters and radians are directly comparable after all. Thus using no scaling factors for the pose parameters makes some sense, and may explain why the general disregard for scaling in the robotics community has not had more consequences. If linkages are much smaller (like fingers) or much larger (like excavators) then the situation is different.

A more general way of choosing the weighting matrix \mathbf{W} is to use a priori information about acceptable relative errors. Such information might arise from specifications of the measurement apparatus. Suppose that the output variables are subject to independent Gaussian noise, such that σ_j^y is the standard deviation in task variable measurement component Δy_j^i , for $j = 1, \dots, m$ pose components. Then the individual diagonal weights are $w_{jj} = 1/\sigma_j^y$, and define

$$\begin{aligned} \mathbf{R}^i &= \text{diag}[(\sigma_1^y)^2, \dots, (\sigma_m^y)^2], \\ \mathbf{R} &= \text{diag}(\mathbf{R}^1, \dots, \mathbf{R}^P), \end{aligned}$$

where the weighting matrix $\mathbf{W} = \mathbf{R}^{-1}$, and \mathbf{R} is called the *covariance matrix*.

The weighted least-squares estimate is

$$\Delta\phi = (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{R}^{-1} \Delta\mathbf{y}. \quad (14.81)$$

The resulting scaled output variable $\Delta\tilde{y}_j^l = \Delta y_j^l / \sigma_j^y$ is dimensionless. The larger the uncertainty σ_j^y , the less this variable influences the least-squares solution relative to other variables. The standard deviations σ_j^y are not necessarily the same as endpoint measurement accuracy, because model errors and input noise also contribute to output errors.

The weighted least-squares solution using standard deviations is variously called the Gauss–Markov estimate, the generalized least-squares estimate, or the best linear unbiased estimator (BLUE) [14.3]. It is the minimum covariance estimate (on the parameter error) of all unbiased estimators. A significant point is that the standard deviations of the scaled components of $\Delta\tilde{\mathbf{y}}$ are all about the same size, or the covariance $\tilde{\mathbf{R}} = \text{cov}(\Delta\tilde{\mathbf{y}}) = \mathbf{I}$, the identity matrix. Hence the Euclidean norm of the error vector $\Delta\tilde{\mathbf{y}}$ is a reasonable measure of its size.

Often we do not know the covariance matrix \mathbf{R} that well. An estimate of the standard deviations after running the calibration procedure is obtained from the χ^2 statistic [14.38, 40]:

$$\chi^2 = (\Delta\mathbf{y} - \mathbf{A}\Delta\phi)^\top \mathbf{R}^{-1} (\Delta\mathbf{y} - \mathbf{A}\Delta\phi). \quad (14.82)$$

This equation is the same as the residual error equation (14.79), with substitution for $\mathbf{W} = \mathbf{R}^{-1}$. χ^2 is nothing more than the weighted residuals after calibration. The expected value of χ^2 is

$$E(\chi^2) \equiv \nu = PK - R, \quad (14.83)$$

where E is the expectation operator. That is to say, the unweighted residuals $(\Delta\mathbf{y} - \mathbf{A}\Delta\phi)^2$ should, with enough measurements, approximate the true covariance. We may now uniformly scale an initial estimate of \mathbf{R} after a preliminary calibration, based on the value of χ^2 ,

$$\hat{\mathbf{R}} = \frac{\chi^2}{\nu} \mathbf{R}, \quad (14.84)$$

where $\hat{\mathbf{R}}$ is the revised estimate of the covariance matrix.

Parameter Scaling

Scaling of parameters is important for proper convergence in nonlinear optimization and for singular value decomposition. If parameters have vastly different magnitudes, then the singular values are not directly comparable. Also, parameter scaling can improve the

conditioning of the regressor \mathbf{A} and help to avoid invertibility problems.

Whereas left multiplication of \mathbf{A} in (14.79) results in task variable scaling, right multiplication of \mathbf{A} by a weighting matrix \mathbf{H} results in parameter scaling [14.39]:

$$\Delta\mathbf{y} = (\mathbf{A}\mathbf{H})(\mathbf{H}^{-1}\Delta\phi) \equiv \bar{\mathbf{A}}\bar{\Delta\phi}, \quad (14.85)$$

where the scaled Jacobian and parameters are $\bar{\mathbf{A}} = \mathbf{A}\mathbf{H}$ and $\bar{\Delta\phi} = \mathbf{H}^{-1}\Delta\phi$. The least-squares solution is not changed by parameter scaling, whereas it is by task variable scaling.

The most common approach towards parameter weighting is column scaling, which does not require a priori statistical information. Define a diagonal matrix $\mathbf{H} = \text{diag}(h_1, \dots, h_{N_{\text{par}}})$ with elements

$$h_j = \begin{cases} \|\mathbf{c}_j\|^{-1} & \text{if } \|\mathbf{c}_j\| \neq 0 \\ 1 & \text{if } \|\mathbf{c}_j\| = 0 \end{cases} \quad (14.86)$$

where \mathbf{a}_j is the j th column of \mathbf{A} . Then (14.85) becomes

$$\Delta\mathbf{y} = \sum_{j=1}^{N_{\text{par}}} \frac{\mathbf{a}_j}{\|\mathbf{a}_j\|} \Delta\phi_j \|\mathbf{a}_j\|. \quad (14.87)$$

Suppose that $\Delta\mathbf{y}$ has been previously normalized; then its length is meaningful. Each $\mathbf{a}_j / \|\mathbf{a}_j\|$ is a unit vector, and hence each scaled parameter $\Delta\phi_j \|\mathbf{a}_j\|$ is about the same size and has the same effect on $\Delta\mathbf{y}$.

Schröder [14.32] identified a problem with column scaling, namely that poor identifiability of parameters can result in very small Euclidean norms, which result in very large scaling factors. This results in strong amplification of uncertainties of \mathbf{A} . Instead, Schröder proposed scaling based on the effect on the anticipated error of the robot (as previously discussed under task variable scaling).

In an ideal case, one would have a priori knowledge of the expected value ϕ_0 of the parameter vector and the standard deviation σ_j^ϕ of each of the parameter vector components. More generally, the parameter distribution would be described by a covariance matrix \mathbf{M} , but usually this information is not available. Instead, the estimate $\hat{\mathbf{M}}$ of the covariance from (14.69) can be used.

If the output measurement covariance \mathbf{R}^{-1} and parameter error covariance \mathbf{M} are both available, one may define a new least-squares optimization criterion that combines output errors with parameter errors to yield a new χ^2 statistic:

$$\chi^2 = (\Delta\mathbf{y} - \mathbf{A}\Delta\phi)^\top \mathbf{R}^{-1} (\Delta\mathbf{y} - \mathbf{A}\Delta\phi) + \Delta\phi^\top \mathbf{M}^{-1} \Delta\phi. \quad (14.88)$$

Its solution is the minimum-variance estimate, which unlike (14.81) is biased

$$\Delta\phi = (\mathbf{A}^\top \mathbf{R}^{-1} \mathbf{A} + \mathbf{M}^{-1})^{-1} \mathbf{A}^\top \mathbf{R}^{-1} \Delta\mathbf{y}. \quad (14.89)$$

The Kalman filter recursively solves the same problem [14.55, 56]; when the state is nonvarying, there is a constant process, and there is no process

noise [14.13, 57]. The Gauss–Markov estimate is the limiting case when \mathbf{M}^{-1} is zero, i.e., there is no a priori information about the parameters. Again, there is an issue of determining the covariances. As for the Gauss–Markov estimate, the expected value of χ^2 can be used to uniformly scale \mathbf{R} and \mathbf{M} *post facto* [14.58].

14.5 Conclusions and Further Reading

This chapter has presented methods for calibrating the kinematic parameters and the inertial parameters of robot manipulators. Both methods are instances of parameter estimation using least squares. Inertial parameters appear linearly in the equations of motion, and ordinary least squares can be used. Kinematic parameters appear nonlinearly because of sines and cosines, and so nonlinear estimation via the Gauss–Newton method can be applied.

There are domain-specific issues for setting up the calibration equations. For kinematic calibration, Hayati parameters have to be mixed with Denavit–Hartenberg parameters to handle the case of nearly parallel joint axes. The calibration equations have to take into account how the endpoint is measured or constrained. By examining in a detailed fashion the possible joint sequences, including parallel or prismatic joints, a minimal parameterization is achieved that obviates problems of identifiability.

The calibration index has been presented as categorizing all kinematic calibration methods. This index counts the number of equations generated per pose, by calculating the excess of sensing over mobility. A key is that all calibration methods can be viewed as closed-loop methods, where any endpoint sensing system is considered as a joint. Parallel robots are handled by incorporating multiple closed loops.

For inertial parameter estimation, the link estimation problem has been viewed as an instance of load estimation but with restricted sensing and motion at the joints. The recursive Newton–Euler equations lead to a formulation of the regressor matrix as an upper-diagonal matrix. A minimal parameterization is straightforward for serial and spanning robots.

Numerical methods were presented to handle unidentifiable parameters if a minimal parameterization has not been achieved. These methods hinge on

the singular value or QR decomposition of the regressor matrix. The singular values can be examined to decide which parameters are not identifiable and should be eliminated. Alternatively, a small singular value can simply be zeroed to eliminate the effects of poorly identifiable parameters without explicitly eliminating them. The former yields a structural model, while the latter yields a prediction model. A priori estimates of parameters can be taken into account by using damped least squares.

The adequacy of the measurement set for parameter estimation is captured by different proposals for an observability index. This index can be related to the literature on alphabet optimalities in experimental design.

Finally, the scaling of measurements or of parameters is important for a well-conditioned numerical estimation, and is essential in order to compare singular values. When uncertainties in measurements and in parameters are included as weights, the optimal minimum variance estimate can be found, which is related to the Kalman filter. If not known beforehand, these uncertainties can be estimated from the data.

14.5.1 Relation to Other Chapters

Estimation involving least squares and Kalman filtering is discussed in Chap. 4 in a similar context. The problem of estimating properties of the world through sensors is very similar to model identification. Recursive estimation techniques are particularly appropriate when a robot needs to update its world model incrementally. For model identification, employing a recursive formulation is not particularly helpful, as the machinery of recursive updating can obscure numerical issues with the total data.

The singular value decomposition appears in other contexts. Chapters 10 and 11 analyze equal motion capability in different directions by measures similar to the observability indexes: O_1 has a counterpart in the manipulability measure, and O_2 the condition number and O_3 the minimum singular value appear again. By contrast, the concern in calibration is good data in all directions as captured by the singular values. Chapter 11 employs the singular value decomposition to analyze redundant mechanisms. Whereas parameter estimation generally is an overconstrained least-squares problem (many more measurements than parameters), redundant mechanisms are underconstrained (more joint angles than task variables). Instead of signalling identifiability problems, zero singular values indicate the null space of the Jacobian. Damped least squares is used in Chap. 11 to avoid singularities. Just as the true parameters are perturbed by damped least squares in calibration, the trajectory is perturbed in order to get around a numerical conditioning issue.

Sensors involved in positioning a robot can have aspects of their sensor models calibrated as well, such as the gain of a potentiometer. Camera calibration is discussed in Chaps. 23 and 24. Camera models can be determined at the same time as the kinematic models of robot manipulators [14.33, 59], including intrinsic camera parameters such as the pinhole model discussed in Chap. 4 as well as the extrinsic parameters of where the camera is located.

References

- 14.1 P.Z. Marmarelis, V.Z. Marmarelis: *Analysis of Physiological Systems* (Plenum, London 1978)
- 14.2 C.W. Wampler, J.M. Hollerbach, T. Arai: An implicit loop method for kinematic calibration and its application to closed-chain mechanisms, *IEEE Trans. Robot. Autom.* **11**, 710–724 (1995)
- 14.3 J.P. Norton: *An Introduction to Identification* (Academic, London 1986)
- 14.4 G.H. Golub, C.F. Van Loan: *Matrix Computations* (Johns Hopkins Univ. Press, Baltimore 1989)
- 14.5 H. West, E. Papadopoulos, S. Dubowsky, H. Cheah: A method for estimating the mass properties of a manipulator by measuring the reaction moments at its base, *Proc. IEEE Int. Conf. Robot. Autom.*, Scottsdale (IEEE Computer Society Press, Washington 1989) pp. 1510–1516
- 14.6 R.P. Paul: *Robot Manipulators: Mathematics, Programming, and Control* (MIT Press, Cambridge 1981)
- 14.7 S.A. Hayati, M. Mirmirani: Improving the absolute positioning accuracy of robot manipulators, *J. Robot. Syst.* **2**, 397–413 (1985)
- 14.8 J.M. Hollerbach, C.W. Wampler: The calibration index and taxonomy of kinematic calibration methods, *Int. J. Robot. Res.* **15**, 573–591 (1996)
- 14.9 A. Goswami, A. Quaid, M. Peshkin: Identifying robot parameters using partial pose information, *IEEE Contr. Syst.* **13**, 6–14 (1993)
- 14.10 M.R. Driels, W.E. Swayze: Automated partial pose measurement system for manipulator calibration experiments, *IEEE Trans. Robot. Autom.* **10**, 430–440 (1994)
- 14.11 G.-R. Tang, L.-S. Liu: Robot calibration using a single laser displacement meter, *Mechatronics* **3**, 503–516 (1993)
- 14.12 D.E. Whitney, C.A. Lozinski, J.M. Rourke: Industrial robot forward calibration method and results, *ASME J. Dyn. Syst. Meas. Contr.* **108**, 1–8 (1986)
- 14.13 B.W. Mooring, Z.S. Roth, M.R. Driels: *Fundamentals of Manipulator Calibration* (Wiley Interscience, New York 1991)
- 14.14 K. Lau, R. Hocken, L. Haynes: Robot performance measurements using automatic laser tracking

14.5.2 Further Reading

Screw Axis Measurement

An alternative to nonlinear least squares to estimate kinematic parameters is a class of methods which measure the joint axes as lines in space, termed *screw axis measurement* in [14.8]. One approach is *circle point analysis*, which involves moving one joint at a time to generate a circle at a distal measurement point [14.13]. Other approaches measure the Jacobian matrix, which contains as columns the joint screws [14.60]. With knowledge of the joint axes, the kinematic parameters are straightforwardly extracted without nonlinear search. The accuracy of this class of methods may not be as good as the nonlinear least-squares methods.

Total Least Squares

Ordinary least squares assumes that there is only noise in the output measurements, but often there is noise in the inputs as well. Input noise is known to lead to bias errors [14.3]. A framework for handling both input and output noise is total least squares [14.61], also known as orthogonal distance regression [14.62] or errors-in-variables regression [14.63]. Nonlinear total least squares has been applied to robot calibration in [14.2, 64, 65]. In the implicit loop method of [14.2], by treating endpoint measurements equally with joint measurements, no distinction is made between input and output noise.

- techniques, *Robot. Comput.-Integr. Manuf.* **2**, 227–236 (1985)
- 14.15 C.H. An, C.H. Atkeson, J.M. Hollerbach: *Model-Based Control of a Robot Manipulator* (MIT Press, Cambridge 1988)
- 14.16 M. Vincze, J.P. Prenninger, H. Gander: A laser tracking system to measure position and orientation of robot end effectors under motion, *Int. J. Robot. Res.* **13**, 305–314 (1994)
- 14.17 J.M. McCarthy: *Introduction to Theoretical Kinematics* (MIT Press, Cambridge 1990)
- 14.18 D.J. Bennet, J.M. Hollerbach: Autonomous calibration of single-loop closed kinematic chains formed by manipulators with passive endpoint constraints, *IEEE Trans. Robot. Autom.* **7**, 597–606 (1991)
- 14.19 W.S. Newman, D.W. Osborn: A new method for kinematic parameter calibration via laser line tracking, *Proc. IEEE Int. Conf. Robot. Autom.*, Atlanta, Vol. 2 (IEEE Computer Society Press, Washington 1993) pp. 160–165
- 14.20 X.-L. Zhong, J.M. Lewis: A new method for autonomous robot calibration, *Proc. IEEE Int. Conf. Robot. Autom.*, Nagoya (IEEE Computer Society Press, Washington 1995) pp. 1790–1795
- 14.21 J.M. Hollerbach, D.M. Lokhorst: Closed-loop kinematic calibration of the RSI 6-DOF hand controller, *IEEE Trans. Robot. Autom.* **11**, 352–359 (1995)
- 14.22 A. Nahvi, J.M. Hollerbach, V. Hayward: Closed-loop kinematic calibration of a parallel-drive shoulder joint, *Proc. IEEE Int. Conf. Robot. Autom.*, San Diego (IEEE Computer Society Press, Washington 1994) pp. 407–412
- 14.23 O. Masory, J. Wang, H. Zhuang: On the accuracy of a Stewart platform – part II Kinematic calibration and compensation, *Proc. IEEE Int. Conf. Robot. Autom.*, Atlanta (IEEE Computer Society Press, Washington 1994) pp. 725–731
- 14.24 M. Gautier: Dynamic identification of robots with power model, *Proc. IEEE Int. Conf. Robot. Autom.* (Albuquerque 1997) pp. 1922–1927
- 14.25 B. Armstrong-Helouvry: *Control of Machines with Friction* (Kluwer Academic, Boston 1991)
- 14.26 B. Armstrong-Helouvry, P. Dupont, C. Canudas de Wit: A survey of models, analysis tools and compensation methods for the control of machines with friction, *Automatica* **30**, 1083–1138 (1994)
- 14.27 F. Aghili, J.M. Hollerbach, M. Buehler: A modular and high-precision motion control system with an integrated motor, *IEEE/ASME Trans. Mechatron.* **12**, 317–329 (2007)
- 14.28 W.S. Newman, J.J. Patel: Experiments in torque control of the Adept One robot, *Proc. IEEE Int. Conf. Robot. Autom.* (Sacramento 1991) pp. 1867–1872
- 14.29 W. Khalil, E. Dombre: *Modeling, Identification and Control of Robots* (Taylor Francis, New York 2002)
- 14.30 W. Khalil, O. Ibrahim: General solution for the dynamic modeling of parallel robots, *J. Intell. Robot. Syst.* **49**, 19–37 (2007)
- 14.31 S. Guegan, W. Khalil, P. Lemoine: Identification of the dynamic parameters of the Orthoglide, *Proc. IEEE Int. Conf. Robot. Autom.* (Taiwan 2003) pp. 3272–3277
- 14.32 K. Schroer: Theory of kinematic modelling and numerical procedures for robot calibration. In: *Robot Calibration*, ed. by R. Bernhardt, S.L. Albright (Chapman Hall, London 1993) pp. 157–196
- 14.33 H. Zhuang, Z.S. Roth: *Camera-Aided Robot Calibration* (CRC, Boca Raton 1996)
- 14.34 J.J. Dongarra, C.B. Mohler, J.R. Bunch, G.W. Stewart: *LINPACK User's Guide* (SIAM, Philadelphia 1979)
- 14.35 M. Gautier, W. Khalil: Direct calculation of minimum set of inertial parameters of serial robots, *IEEE Trans. Robot. Autom.* **RA-6**, 368–373 (1990)
- 14.36 W. Khalil, F. Bennis: Symbolic calculation of the base inertial parameters of closed-loop robots, *Int. J. Robot. Res.* **14**, 112–128 (1995)
- 14.37 W. Khalil, S. Guegan: Inverse and direct dynamic modeling of Gough–Stewart robots, *Trans. Robot. Autom.* **20**, 754–762 (2004)
- 14.38 W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery: *Numerical Recipes in C* (Cambridge Univ. Press, Cambridge 1992)
- 14.39 C.L. Lawson, R.J. Hanson: *Solving Least Squares Problems* (Prentice Hall, Englewood Cliffs 1974)
- 14.40 P.R. Bevington, D.K. Robinson: *Data Reduction and Error Analysis for the Physical Sciences* (McGraw-Hill, New York 1992)
- 14.41 B. Armstrong: On finding exciting trajectories for identification experiments involving systems with nonlinear dynamics, *Int. J. Robot. Res.* **8**, 28–48 (1989)
- 14.42 J. Fiefer, J. Wolfowitz: Optimum designs in regression problems, *Ann. Math. Stat.* **30**, 271–294 (1959)
- 14.43 Y. Sun, J.M. Hollerbach: Observability index selection for robot calibration, *Proc. IEEE Int. Conf. Robot. Autom.* (Pasadena 2008), submitted
- 14.44 J.H. Borm, C.H. Menq: Determination of optimal measurement configurations for robot calibration based on observability measure, *Int. J. Robot. Res.* **10**, 51–63 (1991)
- 14.45 C.H. Menq, J.H. Borm, J.Z. Lai: Identification and observability measure of a basis set of error parameters in robot calibration, *ASME J. Mechamissions Autom. Des.* **111**(4), 513–518 (1989)
- 14.46 M. Gautier, W. Khalil: Exciting trajectories for inertial parameter identification, *Int. J. Robot. Res.* **11**, 362–375 (1992)
- 14.47 M.R. Driels, U.S. Pathre: Significance of observation strategy on the design of robot, *J. Robot. Syst.* **7**, 197–223 (1990)

- 14.48 A. Nahvi, J.M. Hollerbach: The noise amplification index for optimal pose selection in robot calibration, Proc. IEEE Int. Conf. Robot. Autom. (1996) pp. 647–654
- 14.49 D. Daney, B. Madeline, Y. Papegay: Choosing measurement poses for robot calibration with local convergence method and Tabu search, Int. J. Robot. Res. **24**(6), 501–518 (2005)
- 14.50 Y. Sun, J.M. Hollerbach: Active robot calibration algorithm, Proc. IEEE Int. Conf. Robot. Autom. (Pasadena 2008), submitted
- 14.51 T.J. Mitchell: An algorithm for the construction of D-Optimal experimental designs, Technometrics **16**(2), 203–210 (1974)
- 14.52 J. Swevers, C. Ganseman, D.B. Tukul, J. De Schutter, H. Van Brussel: Optimal robot excitation and identification, IEEE Trans. Robot. Autom. **13**, 730–740 (1997)
- 14.53 P.O. Vandanjon, M. Gautier, P. Desbats: Identification of robots inertial parameters by means of spectrum analysis, Proc. IEEE Int. Conf. Robot. Autom. (Nagoya 1995) pp. 3033–3038
- 14.54 E. Walter, L. Pronzato: *Identification of Parametric Models from Experimental Data* (Springer, London 1997)
- 14.55 D.G. Luenberger: *Optimization by Vector Space Methods* (Wiley, New York 1969)
- 14.56 H.W. Sorenson: Least-squares estimation: from Gauss to Kalman, IEEE Spectrum **7**, 63–68 (1970)
- 14.57 Z. Roth, B.W. Mooring, B. Ravani: An overview of robot calibration, IEEE J. Robot. Autom. **3**, 377–386 (1987)
- 14.58 A.E. Bryson Jr., Y.-C. Ho: *Applied Optimal Control* (Hemisphere, Washington 1975)
- 14.59 D.J. Bennet, J.M. Hollerbach, D. Geiger: Autonomous robot calibration for hand-eye coordination, Int. J. Robot. Res. **10**, 550–559 (1991)
- 14.60 D.J. Bennet, J.M. Hollerbach, P.D. Henri: Kinematic calibration by direct estimation of the Jacobian matrix, Proc. IEEE Int. Conf. Robot. Autom. (Nice 1992) pp. 351–357
- 14.61 S. Van Huffel, J. Vandewalle: *The Total Least Squares Problem: Computational Aspects and Analysis* (SIAM, Philadelphia 1991)
- 14.62 P.T. Boggs, R.H. Byrd, R.B. Schnabel: A stable and efficient algorithm for nonlinear orthogonal distance regression, SIAM J. Sci. Stat. Comput. **8**, 1052–1078 (1987)
- 14.63 W.A. Fuller: *Measurement Error Models* (Wiley, New York 1987)
- 14.64 J.-M. Renders, E. Rossignol, M. Becquet, R. Hanus: Kinematic calibration and geometrical parameter identification for robots, IEEE Trans. Robot. Autom. **7**, 721–732 (1991)
- 14.65 G. Zak, B. Benhabib, R.G. Fenton, I. Saban: Application of the weighted least squares parameter estimation method for robot calibration, J. Mech. Des. **116**, 890–893 (1994)

Robot Hands

15. Robot Hands

Claudio Melchiorri, Makoto Kaneko

Multifingered robot hands have a potential capability for achieving dexterous manipulation of objects by using rolling and sliding motions. This chapter addresses design, actuation, sensing and control of multifingered robot hands. From the design viewpoint, they have a strong constraint in actuator implementation due to the space limitation in each joint. After briefly introducing the overview of anthropomorphic end-effector and its dexterity in Sect. 15.1, various approaches for actuation are provided with their advantages and disadvantages in Sect. 15.2. The key classification is (1) remote actuation or build-in actuation and (2) the relationship between the number of joints and the number of actuator. In Sect. 15.3, actuator and sensors used for multifingered hands are described. In Sect. 15.4, modeling and control are introduced by considering both dynamic effects and friction. Applications and trends are given in Sect. 15.5. Finally, this chapter is closed with conclusions and further reading.

15.1 Basic Concepts	346
15.1.1 Anthropomorphic End-Effectors	346
15.1.2 Dexterity of a Robotic Hand	347
15.2 Design of Robot Hands	347
15.2.1 Actuator Placement and Motion Transmission	347
15.2.2 Actuation Architectures	348
15.3 Technologies for Actuation and Sensing .	351
15.3.1 Actuation	351
15.3.2 Sensors	351
15.4 Modeling and Control of a Robot Hand ...	355
15.4.1 Dynamic Effects of Flexible Transmission Systems	355
15.4.2 Transmission Model of Tendon-Outer-Tube System	357
15.4.3 Control Through Single-Acting Actuators	358
15.4.4 Control of a Robot Hand	358
15.5 Applications and Trends	359
15.6 Conclusions and Further Reading	359
References	359

Human hands have great potentiality not only for grasping objects of various shapes and dimensions, but also for manipulating them in a dexterous manner. It is common experience that, by training, one can perform acrobatic manipulation of stick-shaped objects, manipulate a pencil by using rolling or sliding motions, and perform precise operations requiring fine control of small tools or objects. It is obvious that this kind of dexterity cannot be achieved by a simple gripper capable of open/close motion only. A multifingered robot hand can therefore provide a great opportunity for achieving such a dexterous manipulation in a robotic system. Moreover, we have also to consider that human beings do not use hands only for grasping or manipulating objects. Exploration, touch, and perception of physical properties

(roughness, temperature, weight, to mention just a few) are other fundamental tasks that we are usually able to perform thanks to our hands. We expect this type of capabilities also from robotic end-effectors and therefore, by adding quite advanced sensing equipments and proper control strategies, we may improve the interaction capabilities with the environment, achieving, for example, active exploration, detection of sensing surface properties (local friction, impedance, and so on), tasks that are usually very hard or impossible for simple grippers. For these and other reasons the study of multifingered robot hands has greatly interested the research community since the early days of robotics.

It was in late 1970s that *Okada* developed a multifingered robot hand with a tendon driving system

and demonstrated a nut-opening motion [15.1]. In the early 1980s, two major projects on multifingered robot hands have been launched: the Stanford/Jet Propulsion Laboratory (JPL) hand and the Massachusetts Institute of Technology (MIT)/Utah hand [15.2, 3]. These two robot hands still represent a milestone and a term of comparison for the design of new devices. Since then, several multifingered hands have been designed and developed in a number research institutes all over the world. Among the most known, one can mention the Deutsches Zentrum für Luft- und Raumfahrt (DLR) hand, the Mechanical Engineering Laboratory (MEL) hand, the Electro-

Technical Laboratory (ETL) hand, the Darmstadt hand, the Karlsruhe hand, the Bologna hand, the Barrett hand, the Yasukawa hand, the Gifu hand, the U-Tokyo-hand, the Hiroshima hand, and many others [15.4–8].

When designing a multifingered hand, on the basis of its utilization, one should first define the following key issues: number and kinematic configuration of the fingers, anthropomorphic or nonanthropomorphic aspect, built-in or remote actuation, transmission system (in case of remote actuation), sensor assignment, integration with a carrying device (robot arm), and control. All these aspects are considered in this chapter.

15.1 Basic Concepts

Before illustrating the main issues involved in the design and use of a robotic hand, it is necessary to discuss some basic concepts and definitions often encountered when dealing with these devices. In particular, terms like *dexterity* and *anthropomorphism* must be defined, and their implications on robotic hand design specified.

15.1.1 Anthropomorphic End-Effectors

The term *anthropomorphism* denotes the capability of a robotic end-effector to mimic the human hand, partly or totally, as far as shape, size, consistency, and general aspect (including color, temperature, and so on) are considered. As the word itself suggests, anthropomorphism is related to the external perceivable properties, and is not, itself, a measure of what the hand can do. On the contrary, *dexterity* is related to actual functionality and not to shape or aesthetic factors. In this sense anthropomorphism and dexterity are *orthogonal* concepts, whose reciprocal dependence (at least in the robotic field) has been not proved yet.

In fact, we can find in the literature anthropomorphic end-effectors with very poor dexterity level, even though they are called *hands*, as the tasks they can perform are limited to very rough grasping procedures [15.9]. Similarly, we can find smart end-effectors, capable of sophisticated manipulation procedures, without any level of anthropomorphism, e.g., the DxGrip-II [15.10]. Anthropomorphism itself is neither necessary nor sufficient to achieve dexterity, although it is quite evident that the human hand achieves a very high level of dexterity and represents a preferential paradigm for dexterous robotic manipulation.

Anthropomorphism is a desirable goal in the design of robotic end-effectors mainly for the following reasons:

- the end-effector can operate in a human-oriented environment (e.g., servicing robots), where tasks may be executed either by robots or men;
- the end-effector can be teleoperated by a human operator, by means of special-purpose interfaces (e.g., a data glove), directly reproducing the operator's hand behavior;
- for the purposes of entertainment, assistance, and so on, a human-like aspect and behavior may be specifically required, like for humanoid robots;
- for prosthetic devices anthropomorphism is a quite evident design goal. The development of end-effectors for prosthetic purposes [15.11, 12] has recently produced such advanced devices that they can be fully considered robotic systems.

While it is difficult to quantify the effective degree of dexterity of a robotic system, its anthropomorphism can be defined in a precise and objective way. In particular, the main aspects that contribute to determining the anthropomorphism level of a robotic hand are:

- *Kinematics*: the presence of the main morphological elements (principal upper fingers, secondary upper fingers, opposable thumb, and palm);
- *Contact surfaces*: the extension and smoothness of the contact surfaces, an aspect that reflects on the capability to locate contacts with objects all over the surface of the available links and on the presence of external compliant pads;

- *Size*: both referring to the average size of a human hand and the *correct* size ratio between the links.

15.1.2 Dexterity of a Robotic Hand

Besides the *geometrical* reproduction of the human hand, the main target of research remains the emulation of those functionalities that make it such a versatile end-effector.

Two are the main capabilities of a human hand:

- *prehension*: the ability to grasp and hold objects of different size and shape;
- *apprehension*: the ability to understand through active touch.

In this sense, the human hand is both an *output* and *input* device (see [15.13]). As an output device, it can apply forces in order to obtain stable grasps or perform manipulation procedures. As an input device, it is capable of exploring an unknown environment providing information about the state of the interaction with it. The same features are desirable in robot hands. In fact, the application of robotic systems in unknown environments requires dexterous manipulation abilities to execute complex operations in a flexible way.

A widely accepted definition states that the dexterity of a robotic end-effector is a measure of its capability of changing the configuration of the manipulated object from an initial configuration to a final one, arbitrarily chosen within the device workspace. Generally speak-

ing, with the term *dexterity* we intend the capability of the end-effector, operated by a suitable robotic system, to perform autonomously tasks with a certain level of complexity. An exhaustive review of scientific work developed so far about the dexterity of robotic hands, with a quite complete and up-to-date list of references, can be found in [15.14].

Even though the word dexterity itself has a very positive meaning, it may be useful to consider different levels of dexterity, associated with the growing complexity and criticality of performable tasks. The dexterity domain for robotic hands can be roughly divided in two main areas: *grasping* and *internal manipulation*.

Grasping means the capability to constrain objects in a configuration that is substantially invariant with time (the object is fixed with respect to the hand).

Internal manipulation is controlled motion of the grasped object in the hand workspace, with the constraint configuration changing with time.

Further subdivisions of these two domains have been widely discussed in the literature (different grasp topologies [15.15] and different internal manipulation modes based on internal mobility and/or contact sliding or rolling [15.14]).

Although the notion of dexterity is settled, the way to achieve it remains debated. Factors affecting the actual capabilities of a robotic end-effector are so many that often the analysis and above all the synthesis of dexterous hands do not take into proper consideration some of these elements, namely: morphological features, sensory equipment, control algorithms, task planning strategies, and so on.

15.2 Design of Robot Hands

The mechanical design of an articulated robotic hand can be performed according to many possible design concepts and options, even if a kinematical architecture has already been defined and size and shape specifications imposed. One of the main issues is the design of a proper actuation and transmission system. This aspect is crucial because space and dimensions are usually limited, being in general an anthropomorphic aspect and dimension a design goal to be pursued.

Note that, since many solutions and operating concepts can be adopted, what is presented here aims only to illustrate the most significant solutions, and does not pretend to be a complete discussion of all the possible choices.

15.2.1 Actuator Placement and Motion Transmission

In order to actuate the joints of a robot hand, two basic approaches for the placement of the actuators are possible:

- placing the motors as close as possible to each joint, directly in the fingers and sometimes integrating them within the joint itself;
- placing the motors into the palm or in the forearm; in this case motion is transmitted to each joint by means of (complex) kinematic chains.

In-Site Actuation

In-site actuation can be defined as the case in which the actuator is hosted inside one of the two links connected by the actuated joint or is placed directly inside the joint:

- *Direct-drive actuation*: the actuator is placed directly on the joint, without transmission elements.
- *Link-hosted actuation*: the actuator is placed inside one of the two links constituting the actuated kinematic chain.

In-site actuation simplifies the mechanical configuration of the joint, reducing the complexity of the transmission chain. In particular, it has the great advantage that the motion of the joint is kinematically independent with respect to other joints. Usually, the size of the finger is imposed by the dimension of the actuators, and for technological reasons it is quite difficult to obtain both an anthropomorphic size and the same grasp strength of the human hand. Furthermore, the motors occupy a large space inside the finger structure, making it difficult to host other elements, like sensors or compliant skin layers. A further negative aspect is that, since the mass of the actuators is concentrated inside the finger, the dynamic behavior of the system and its response bandwidth are reduced.

Nevertheless, the recent advancement of actuator technology enables us to directly implement a quite powerful actuator with reasonable size in each joint. This built-in actuation has been adopted, e.g., for the DLR hand [15.4, 16], the ETL hand, the Karlsruhe hand, the Yasukawa hand, the Barrett hand, the Gifu hand, the U-Tokyo hand, and the Hiroshima hand. Since this actuation does not include compliant elements such as tendons, we can keep a stiff transmission system, which leads to a stable control system even under a high gain, see Sect. 15.4. An issue is the routing of wires for both power and signal cables. This issue is more serious in distal joints than for the base joint, since the cables in distal joints produce a relatively large torque disturbance on the first joint, and therefore it is difficult to achieve precise torque control for this joint.

Remote Actuation

Remote actuation is an alternative solution to in-site actuation. In remote actuation, the joint is driven by actuators placed outside the links connected by the joint itself. Remote actuation requires a motion transmission system, which must pass through the joints between the motor and the actuated joint. In some way, remote actuation must consider the problem of kinematic coupling between the actuated joint and the previous ones. Re-

ote actuation is prevalent in biological structures (e.g., in the human hand), where the finger joints are moved by muscles placed on the palm or in the forearm. This human-like approach has been adopted in projects of robotic hands like the University of Bologna (UB) hand or the National Aeronautics and Space Agency (NASA) Robonaut hand [15.17, 18].

Remote actuation systems can be classified according to the type of adopted transmission elements, i.e., flexible- or rigid-link transmission.

Flexible-Link Transmission. Flexible-link transmission is based on deformable connections, either flexible or rotational, that can adapt to variations of configuration by changing the transmission path. Linear flexible transmissions are based on flexible elements with translating motion, subject to tension (more frequently) or tension and compression. Two further subcategories can be identified: pulley-routed flexible elements (tendons, chains, belts) or sheath-routed flexible elements (mainly tendon-like elements). Rotational flexible transmissions are based on flexible rotary shafts that can transmit rotational motion inside the finger structure to the joint, where a final transforming mechanism (a bevel gear or a worm gear) can be used to actuate the joint.

Rigid-Link Transmission. Rigid-link transmission is mainly based on articulated linkages or on rolling conjugated profiles (mainly gear trains). A further subdivision can be made between parallel and nonparallel axes gear trains, like bevel gears, worm gears, and so on.

15.2.2 Actuation Architectures

Both in-site and remote actuation can be applied according to different types of organization, i.e., by using one or more actuators for each joint and by making these actuators work in different ways.

In general, we can consider an overall number N of joints for the robotic hand (the wrist joints are not considered) and a number M of actuators that are used to drive, directly or indirectly, the joints. According to different concepts of actuation and transmission, three main categories of actuation schemes can be identified:

- $M < N$: some joints are passive, coupled, or underactuated;
- $M = N$: each joint has its own actuator and there are no passive, coupled or underactuated joints;
- $M > N$: more than one actuator is operating on a single joint.

These architectures strongly depend on the type of motors. In particular, it is possible to recognize two main actuation modalities:

- **Single-acting actuators:** each motor can generate a controlled motion in one direction only: return motion in the opposite direction must be obtained by an external action, which can be a passive (e.g., a spring) or an active system (e.g., an antagonistic actuator); this is the case of tendon-based transmission systems;
- **Double-acting actuators:** each motor can generate a controlled motion in both directions and can be used alone to drive the joint or to cooperate with other actuators; in this case the functional redundancy can allow sophisticated drive techniques, like push–pull cooperation.

Each category can be further subdivided. In the following, a brief description of the most frequently adopted schemes is presented.

Single-Acting Actuators with Passive Return Elements

Passive elements, like springs, can store energy during the actuation phase, restituting it during the return stroke, see Fig. 15.1a. This mechanism leads to a simplification of the actuation scheme, but requires mechanically backdrivable actuators. Other possible drawbacks are related to the loss of available power for the grasp and the limited response bandwidth in case of low spring stiffness.

Agonistic–Antagonistic Single-Acting Actuators

Two actuators drive the same joint, acting in opposition in different directions (agonistic–antagonistic couple), see Fig. 15.2b. This solution leads to an N -joint $2N$ -

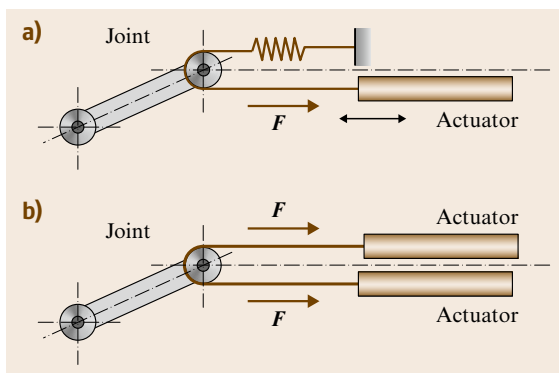


Fig. 15.1 (a) Single-acting actuator with an antagonist passive element and (b) in an agonist–antagonistic configuration

actuators scheme and is quite complex since a large number of actuators must be placed in the hand. On the other hand, it may allow sophisticated control procedures, as both actuators can pull at the same time, with different intensity, generating a driving torque on the joint and a pre-loading of the joint itself (co-contraction, typical of tendon-driven joints).

- **Pros:** co-contraction strategies; possibility to change the joint stiffness according to the grasping phase and therefore to limit the influence of friction during fast approaching motions; independent position/tension control on each actuator can allow compensation of different path length in case of remote transmission; it is the most flexible solution for driving a joint;
- **Cons:** backdrivability of actuators is required; difficulty in hosting two actuators for each joint, both in-site and at a remote location; higher control complexity; greater cost.

Single-Acting Actuators Organized According to the Concept of Actuation Net

This is a very interesting case, mimicking biological systems, but has not yet been implemented in robotic hands, except for some preliminary studies. N joints are driven by M actuators, WITH $N < M < 2N$. Each actuator cooperates in moving more than one joint, thanks to proper net-shaped transmissions.

- **Pros:** co-contraction strategies; possibility to change the joint stiffness according to the grasping phase and therefore to limit the influence of friction during fast approaching motions; reduced number of actuators with respect to the $2N$ -actuator scheme.
- **Cons:** backdrivability of actuators is required; high complexity of the kinematic scheme and therefore high complexity in control.

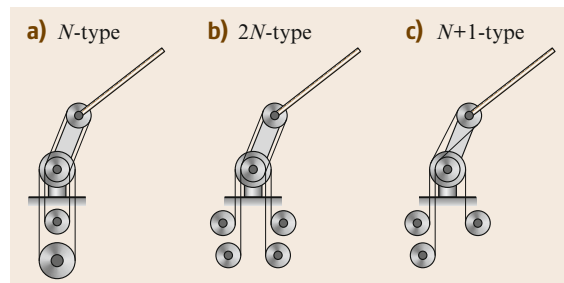


Fig. 15.2a–c Remote actuation (a) N -type (b) $2N$ -type (c) $N+1$ -type

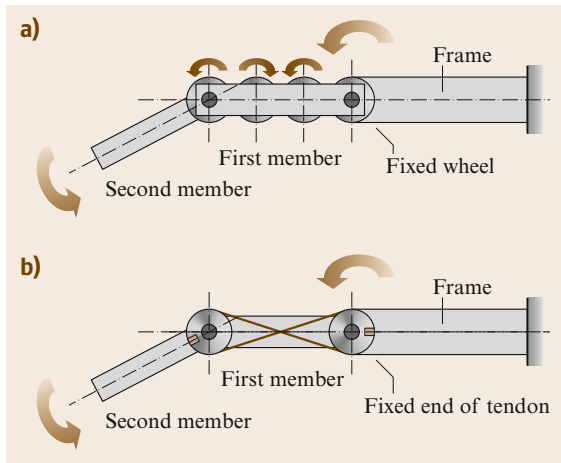


Fig. 15.3 (a) Double-acting actuator with $N = M$ based on gears and (b) tendons

The simplest case of an actuation net is represented by so-called $N+1$ actuation (where N in this case is the number of joints of a finger), frequently adopted in practice, see Fig. 15.2. In this case, all actuators are coupled, and therefore damage to any of them will result in a general failure.

Double-Acting Actuators with $M < N$

In this case, the number of actuators is less than the number of joints. With reference to a single motor and several joints, two main subcases can be defined:

1. the joints are kinematically coupled, in a fixed or variable way, so that the number of degrees of freedom of this subsystems is reduced to one;
2. the joints are selectively actuated by the motor, according to an active or passive selection subsystem.

The former case can be further subdivided.

Joints Kinematically Coupled in a Fixed Way. In this kind of kinematical configuration, each motor can move

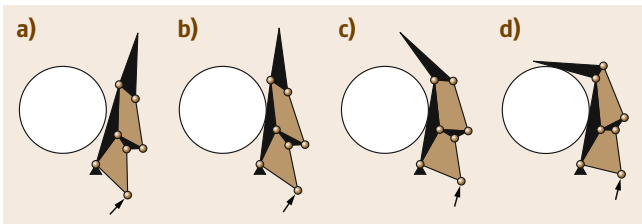


Fig. 15.4 Grasping sequence performed by a finger based on an underactuated mechanism

more joints connected by rigid mechanisms with fixed transmission ratios. A typical application is obtained with the use of a gear train: the first link is directly actuated by a motor, while a gear transmission between a wheel fixed to the frame and a final wheel connected to the joint generates the relative motion of the second link, see Fig. 15.3a. Should the motion of two parallel fingers be required, their connection could be easily obtained by mounting two gear wheels on the same shaft. Another very common way to obtain this kind of kinematical linkage is to use tendon-driven devices, as shown in Fig. 15.3b. In artificial hand design, the main advantage when using joints driven by fixed-ratio mechanisms is the possibility to know and control the position of the second link. A disadvantage is that this kind of mechanisms does not adapt to the shapes of the grasped objects, and this may cause grasp instability.

Joints Coupled in a Nonfixed Way. This is the case of underactuated mechanisms and deformable passive-driven joints. A mechanism is said to be *underactuated* when the number of actuators is smaller than the number of degrees of freedom. When applied to mechanical fingers, this concept may lead to shape adaptation, i. e., underactuated fingers can envelope the objects to be grasped and adapt to their shape even with a reduced number of actuators. In order to obtain a statically determined system, elastic elements and mechanical limits must be introduced in underactuated systems (a simple linear spring are often used). In the case of a finger closing on an object, for instance, the configuration of the finger is determined by the external constraints associated with the object. An example of an underactuated two-degree-of-freedom finger is shown in Fig. 15.4 [15.19]. The finger is actuated through the lower link, and a spring is used to maintain the finger fully extended. A mechanical limit is used to keep the phalanges aligned under the action of this spring when no external forces are applied on the phalanges. Since the joints cannot be controlled independently, the

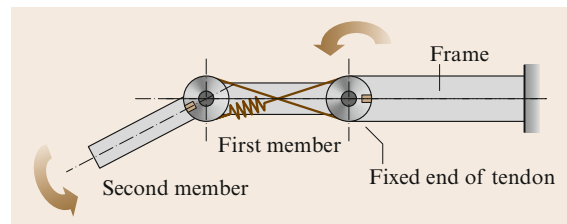


Fig. 15.5 Joints coupled in a nonfixed way

behavior of the finger is determined by the design parameters (i.e., the geometric and the stiffness properties). Hence, the choice of these design parameters is a crucial issue.

Another approach consists of coupling the motion of two adjacent joints by means of deformable linkages. This feature introduces in the kinematical chain the needed compliance to fit to the shapes of the grasped objects. A very simple mechanism of this category is reported in Fig. 15.5. Structurally it is similar to the mechanisms based on a fixed coupling; the only important difference is the addition of a spring to give extensibility to the tendon. This spring allows the motion between the first and second link when an external force is applied to the distal one to be decoupled. This solution is widely used: a well-known example is the DLR hand. The benefits of this solution are mainly due to the possibility to fit to the shapes of objects. A design problem is the choice of the stiffness of the deformable element in order to achieve at the same time a strong grasp and a good shape adaptability.

Joints Selectively Driven by Only One Motor. With this solution, the motion generated by only one (large) motor is transmitted and distributed to several joints. Actuation and control of each joint is obtained by means of

insertion–disinsertion devices like self-acting or commanded clutches.

Double-Acting Actuation, with $M = N$

This is a very common case: each joint is driven in both directions by the same actuator. The achievable performances are therefore similar (equal) in both directions, but particular attention must be paid to backlash, and it is usually necessary to preload the transmission system. In particular, preloading is mandatory in the case of transmission by means of flexible elements such as tendons, see Fig. 15.2a. Furthermore, the adoption of a closed-loop tendon transmission requires that the overall length of the tendon route must be kept constant, according to the concept that wound and unwound parts of the tendon on the motor pulley have the same length; this involves the need for length-compensating mechanisms (e.g., pulley trains, cams) every time that changes in the geometry of the finger cause a differential displacement of the tendons. In spite of this additional complexity, this actuation scheme has been widely used, with simple pulley routing (the UB hand, Okada hand, etc.), or sheath routing (the Salisbury hand, the Dipartimento di Informatica Sistemica e Telematica (DIST) hand), which has a simpler mechanical structure but must face the problem of sheath–tendon friction (the application of a high preload is not convenient in this case).

15.3 Technologies for Actuation and Sensing

In this section, a brief description of the main issues related to technological aspects of actuation and sensing for robot hands is reported. A more general presentation and detailed description of these aspects is given in Part I (Chaps. 3 and 4) and Part C (Chap. 19).

15.3.1 Actuation

Electrical actuators are without doubt the commonest choice for actuating robot hands. As a matter of fact, electric motors have very good performance in terms of position/velocity control, have a reasonable mass/power ratio, and are a very common technology that does not require external devices (such as for hydraulic or pneumatic actuators). However, there are several other possibilities, for example, ultrasonic motors (the Keio hand, [15.20]), chemical actuators, pneumatic actuators as the *McKibben* used in the Shadow hand [15.21]), spring-based actuator (as for the 100G Capturing Robot [15.22]) and others.

In particular, when seeking a quick response, either pneumatic or spring-based actuators may be good solutions, although it should be noted that a braking system with a quick response is essential for achieving good position controllability for this type of actuators.

15.3.2 Sensors

In robot hands, as in other robotic devices, sensors can be classified into two main categories: *proprioceptive* and *exteroceptive* sensors. The first type of sensors measures physical information related to the state of the device itself (e.g., position and velocity), while the second one is devoted to the measurement of data related to the interaction with objects/environment (e.g., applied forces/torques, friction, and shape).

Joint Position/Velocity Sensors

For control purposes, there is the obvious necessity of measuring position/velocity of the actuated joints. A ma-

major problem consists in the limitation of the available space, both for the sensors and for the wires. Different technological solutions can be adopted, but a rather common choice is Hall-effect sensors, which are sufficiently small, precise, and reliable for this type of application. In the case of remote actuation, there is the possibility of having two position/velocity sensors for each joint: one located in the actuator (e.g., an encoder) and one placed in the joint itself, often necessary because of the nonlinearities introduced by the transmission system (elasticity, friction, and so on). Quite often, this latter sensor is specifically designed and implemented for the given hand, as commercially available sensors are too large and not suitable for installation in the joints.

Tendon Tension Sensor and Joint Torque Sensor

It is well known that humans can control fingertip compliance as well as fingertip force by controlling voluntary muscles. In remote actuation, it is essential to measure the tendon tension for two main reasons: to compensate for the friction existing in the transmission system, and to measure the external contact force. Figure 15.6 shows a way of measuring the tendon tension in which the tendon is pressed by an elastic plate with a strain gauge. When tension is applied to the tendon, the sensor measures a force composed of axial and bending force components. The displacement of the elastic plate

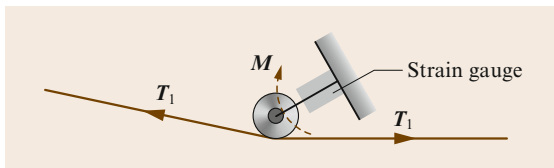


Fig. 15.6 Tendon tension sensing

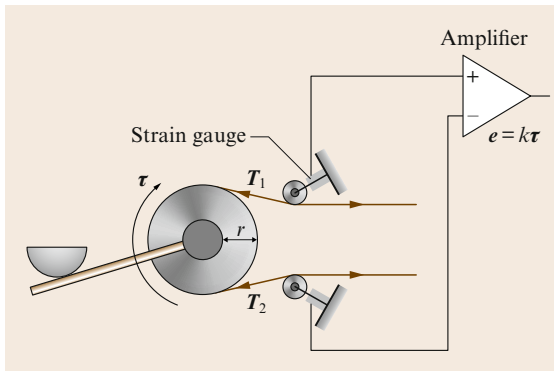


Fig. 15.7 Tension-sensor-based torque sensing

due to the axial force component is negligibly small compared with that due to the bending force component. As a result, the bending force component generates a bending deformation for the elastic plate. This deformation is transformed into an electric signal by the strain gauges attached on the surface of the plate. Now, suppose N -type actuation with two tension sensors, as shown in Fig. 15.7, where joint torque τ is given. Note that $\tau = r(T_1 - T_2)$, where r , T_1 , and T_2 are the pulley radius and tendon tensions, respectively. Since we can measure e_1 and e_2 corresponding to T_1 and T_2 , τ can be obtained by feeding both e_1 and e_2 into the differential circuit. This approach, however, includes a couple of issues. The main problem is the plastic deformation of the sensor plate under an extreme large pre-tension. Once such a plastic deformation has happened, the sensor will never work appropriately anymore. Another minor issue is that two sensors are always necessary for measuring a joint torque. To cope with these issues, the tension-differential-type torque sensor [15.22] can be used, as shown in Fig. 15.8. The sensor is designed with just a single body and partially includes an elastic part where at least one strain gauge is attached. The working principle of the sensor, shown in Fig. 15.8a, supposes that a torque is applied to the joint. This means that T_1 and T_2 have different values. This difference causes a bending force around the strain gauge. The key is that the bending force is kept to zero even under an extremely large tension as long as no joint torque is applied. Therefore,

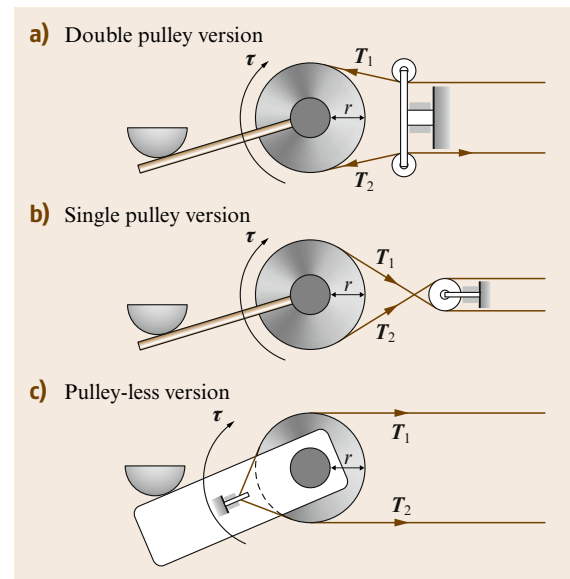


Fig. 15.8 A tension differential type (TDT) sensor [15.22]

we are completely released from the plastic deformation of the elastic plate due to pre-tension. Furthermore, the sensor is constructed with just a single body. There are a couple of variations of this type of torque sensor. When decreasing the pulley distance in Fig. 15.8a, the sensor eventually results in the single-pulley version with zero distance, as shown in Fig. 15.8b. The single-pulley version has been implemented into the Darmstadt hand [15.23] and the MEL hand [15.24]. Furthermore, if the sensor is built into the finger link connected by the concerned tendon, there is no relative motion between the sensor and the tendon. As a result, we can remove the pulley, as shown in Fig. 15.8c. This is called the pulley-less version and has been implemented in the Hiroshima hand. The tension-differential-type torque sensor will be a powerful tool for measuring a tendon drive joint.

Fingertip Tactile (or Force) Sensors

Most robot manipulation and assembly tasks would benefit from the utilization of tactile sensory information. When lifting an object, tactile sensing could detect the onset of slip in time for corrective action to be taken. In addition to the contact point between the fingertip and the object, several object properties, such as the friction coefficient of the object surfaces, surface texture, and weight can be determined by utilizing a fingertip tactile (or force) sensor. A six-axis force sensor allows us to detect contact point as well as the contact force between the finger and the environment if a single contact is assumed. For the finger model shown in Fig. 15.9 we have the following relationship between the sensor output and contact force

$$\mathbf{F}_s = \mathbf{f}, \quad (15.1)$$

$$\mathbf{M}_s = \mathbf{x}_c \times \mathbf{f}, \quad (15.2)$$

where $\mathbf{f} \in \mathbb{R}^3$, $\mathbf{F}_s \in \mathbb{R}^3$, $\mathbf{M}_s \in \mathbb{R}^3$, and $\mathbf{x}_c \in \mathbb{R}^3$ are the external force vector, the force vector measured by the six-axis force sensor, the moment vector measured

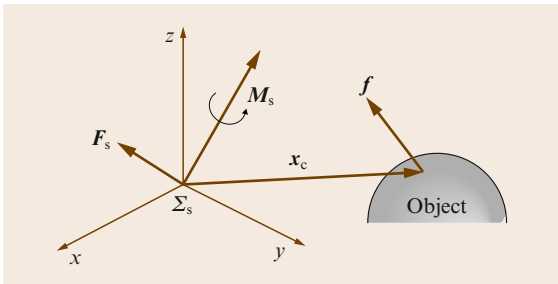


Fig. 15.9 Sensor coordinate system Σ_s

by the six-axis force sensor, and the position vector indicating the contact position, respectively. From the first equation, we can directly obtain the contact force. Putting \mathbf{F}_s into the second equation leads to $\mathbf{M}_s = \mathbf{x}_c \times \mathbf{F}_s$. \mathbf{x}_c is determined in such a way that $\mathbf{M}_s = \mathbf{x}_c \times \mathbf{F}_s$ may be satisfied. For a finger with a convex object, we always have two mathematical solutions, as shown in Fig. 15.10a, where the meaningful solution is the one satisfying $f^n > 0$ (a finger can only push the object). However, for a finger with a concave object, we have at least four mathematical solutions, as shown in Fig. 15.10b, where two of those are physically possible. A finger with a six-axis force sensor located in the fingertip, Fig. 15.10c, can avoid multiple solutions. On the other hand, only forces applied to the fingertip can be detected, and if more links are in contact with the object it would be necessary to have a force/torque sensor placed in each of them.

This type of solution, i.e., a multi-axis sensor for measuring not only forces and torques but also the position of the contact point, is known in the literature as the *intrinsic tactile (IT)* principle [15.25]. In general, compared with the use of traditional tactile sensors (see below) this leads to a simplification in the design since it requires fewer wires and connections for the sensor.

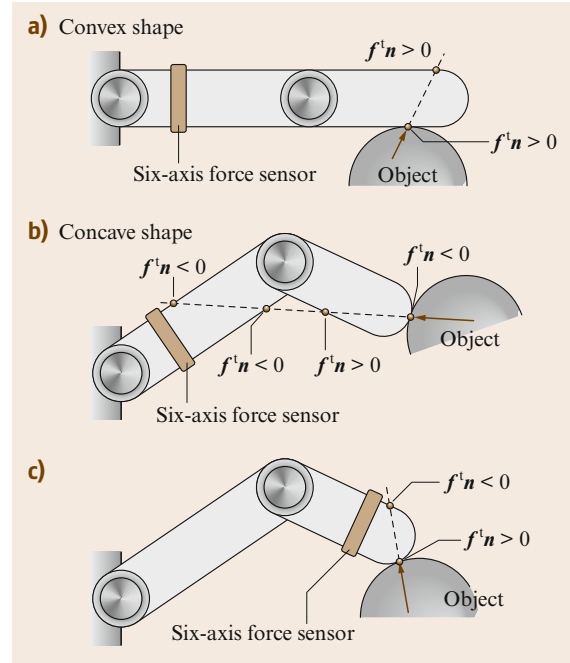


Fig. 15.10a–c Interpretation of solutions: (a) convex shape, (b) concave shape, and (c) six-axis force sensor

Tactile Sensors

Another important class of sensing devices consists of tactile sensors, which are used for several purposes, such as shape recognition, contact point determination, and pressure/force measurement. A number of tactile sensors have been proposed in the literature, with several different solutions concerning the realizable features: optical, piezoresistive, piezoelectric, and so on; see, e.g., [15.26, 27] for an overview on technologies and applications.

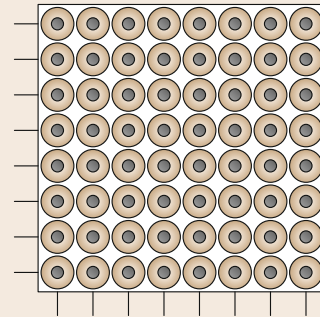
Tactile sensors have been introduced in robotics since the late 1970s. Nowadays, as with force sensors, tactile sensors are also commercially available devices. Probably, they represent the most commonly adopted sensorial class for industrial grippers, even though they are often used as advanced *on/off* devices to check whether a grasp or contact condition occurs.

Usually, they consist of a matrix (array) of sensing elements. Each sensing element is usually referred to as a *taxel* (from *tactile element*), and the whole set of information is called a *tactile image* (Fig. 15.11). The main goal of this class of sensors is to measure the pressure map over the sensing area.

In general, the types of information that may be obtained from a tactile sensor are:

- *contact*: this is the most simple information given by the sensor, concerning the presence or absence of a contact;
- *force*: each sensing element provides information related to the amount of locally applied force, which can be used in several ways for successive elaborations;
- *simple geometrical information*, i. e., the position of the contact area, the geometrical shape of the contact itself (planar, circular, and so on);
- *the main geometrical features of the object*: by proper elaborations of the data from the taxels, it is possible to deduce the type of object in contact with the sensor, for example, a sphere or a cylinder (data relative to the three-dimensional shape);
- *mechanical properties*, such as friction coefficient or roughness. Also thermal properties of the object may be measured by a tactile sensor;
- *slip condition*, i. e., the relative movement between the object and the sensor.

a) Scheme of a tactile sensor



b) Example of data from a tactile sensor

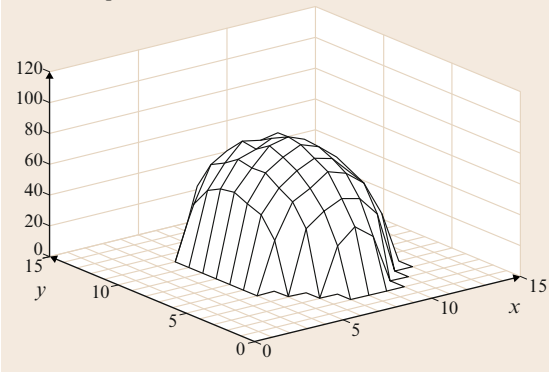


Fig. 15.11a,b A tactile sensor: (a) scheme and (b) example data

Several technologies have been adopted for the design of tactile sensors, ranging from piezoresistive to magnetic, to optical effects, and so on. Among the most common, one can mention:

- resistive and conductive effect
- electromagnetic effect
- capacitive effect
- piezoelectric effect
- optical effect
- mechanical methods

Each of these technologies has positive and negative aspects. Common drawbacks, however, are the size of these sensors, which are usually quite large in comparison with the available space, and the necessity of a large number of electrical connections.

15.4 Modeling and Control of a Robot Hand

The dynamic model of a robot hand with in-site actuation is very similar to the model of a traditional (industrial) robot, and the hand can be considered as a collection of robot manipulators. On the other hand, remote actuation introduces some peculiar features that have to be carefully considered. In particular, the problems linked to nonlinear phenomena (e.g., friction and backlash), the compliance of the transmission system, and the non-collocation of sensors and actuators are critical for the design of the control. Moreover, the use of single-acting actuators, such as tendon-based actuation systems, requires the adoption of proper control techniques that allow the imposition of the desired torque at each joint of the hand despite the coupling among them.

15.4.1 Dynamic Effects of Flexible Transmission Systems

The transmission system of robot hands with remote actuation is usually characterized by a high level of friction and nonnegligible dynamic effects which complicate the control problem. A simple representation considers a single axis motion with two inertial elements linked by an elastic transmission. This is the typical representation of elastic joints in which the former element represents the motor inertia, while the latter is related to the inertial properties of the actuated joint/link, see Fig. 15.12a. More complex models assume a dynamic model for the transmission system, i.e., the

classical representation of tendons based on the serial repetition of masses linked by springs/dampers, reported in Fig. 15.12b. These simple models are particularly useful to understand some drawbacks and limitations due to the fact that actuation system and actuated element are located in two different places and the motion is transmitted by a nonideal (that is, not purely static) element. If we consider the capability of the fingers' joint to apply a force on environment, the effects of the transmission system on the open-loop response of the system modeled as in Fig. 15.12a are a noticeable reduction of the bandwidth and an important phase delay between the input F_a (the force applied by the motor) and the output F_c (the force exchanged at the contact). As shown in Fig. 15.13, the open-loop transfer function

$$\frac{F_a}{F_c} = \frac{(b_c s + k_c)(b_t s + k_t)}{[j_1 s^2 + (b_t + b_c)s + k_t + k_c](j_m s^2 + b_t s + k_t) - (b_t s + k_t)^2} \quad (15.3)$$

is characterized by four poles that, for increasing values of the transmission stiffness k_s , move from their initial locations (which depend on the values of physical parameters j_1 , j_m , etc., although for $k_s = 0$ at least one pole is in the origin of the Gauss plane) towards the poles of a system with an infinitely rigid transmission (for k_c tending to ∞ , two poles go to infinity) and a total inertia given by the contributions of both the

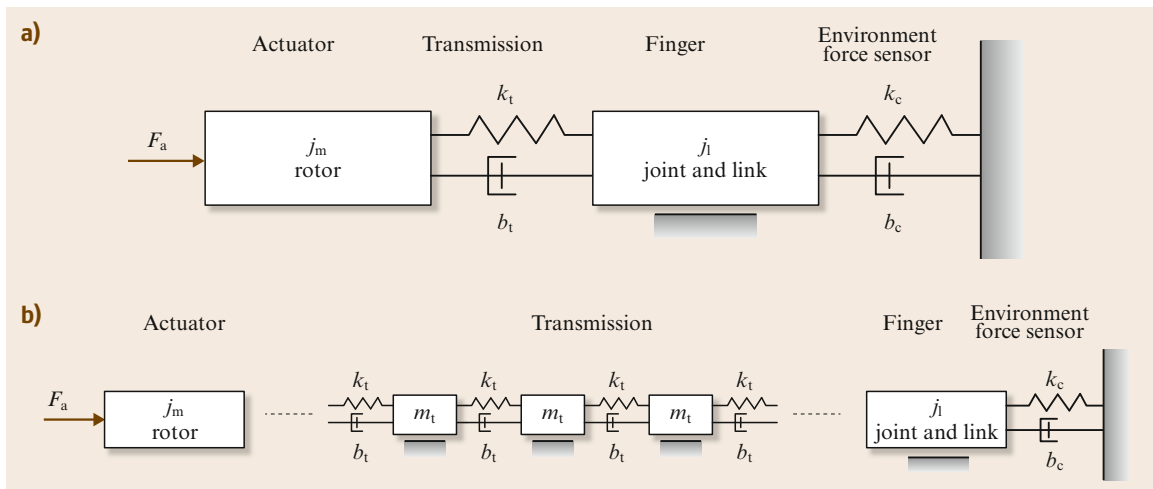


Fig. 15.12 (a) Model of a robot joint with transmission flexibility and (b) with tendon-based transmission

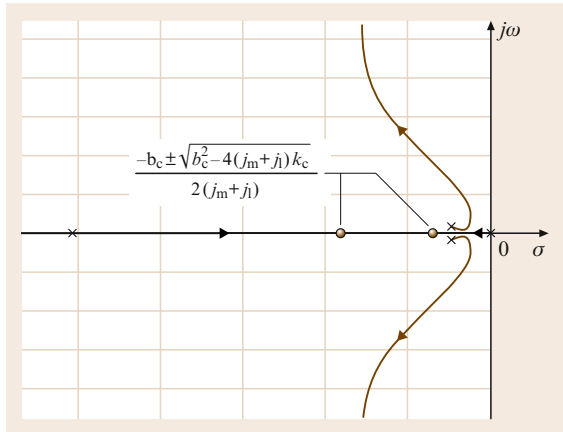


Fig. 15.13 Root contour of the transfer function (15.3) with variable k_t

motor and the link. In this case the transfer function is:

$$\frac{F_a}{F_c} = \frac{(b_c s + k_c)}{(j_l + j_m)s^2 + b_c s + k_c} \quad (15.4)$$

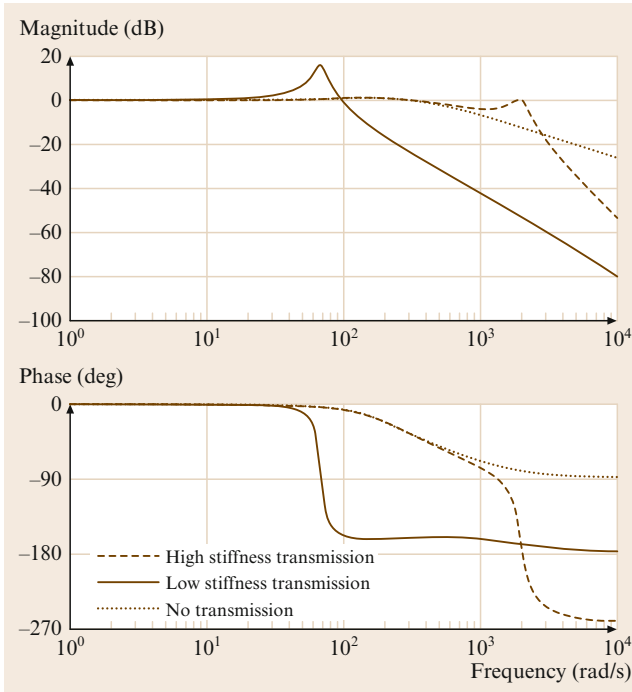


Fig. 15.14 Bode plots of the open-loop transfer function (15.3) for low stiffness values (*continuous line*) high stiffness value (*dashed line*) and with no transmission (*dotted line*)

As a consequence, the bandwidth of the system with flexible transmission, which for high values of k_s approximates those of (15.4), decreases when the compliance of the transmission is not negligible; see the Bode plots reported in Fig. 15.14. The bandwidth of a system with flexible transmission is strongly affected by the location of the speed reducer: when the reducer is placed at the joint the bandwidth is K_r times higher than the one achievable with the reduction applied directly on the motor [15.28] (see Fig. 15.15; K_r is the reduction ratio). Moreover, it is worth noticing that for low values of the stiffness k_t a sharp phase drop occurs at the frequency of the flexible mode. Therefore, there are some (relatively low) frequencies at which the force applied by the motor and the one measured with a sensor in the finger's joint are completely out of phase. These effects, which may cause the instability of the overall system under force control (or impedance control) are referred to as *non-collocation*. In general, when actuators and sensors are physically located at different points of a flexible structure (or a structure with flexible transmission), there will be unstable modes in the closed-loop system [15.29].

From the control viewpoint, the problem of mechanical transmission flexibility is further exasperated by the nonlinear frictional phenomena that inevitably affect remote actuation and motion transmission. In fact, the linear viscous friction, represented in Fig. 15.12 by the damping coefficient b_t , is accompanied by stiction and Coulomb friction, both of which are discontinuous at zero velocity (see Fig. 15.16). These nonlinearities may cause limit cycles and input-dependent stability, and must be accurately taken into account in the design of the robot hand structure as well as of its control

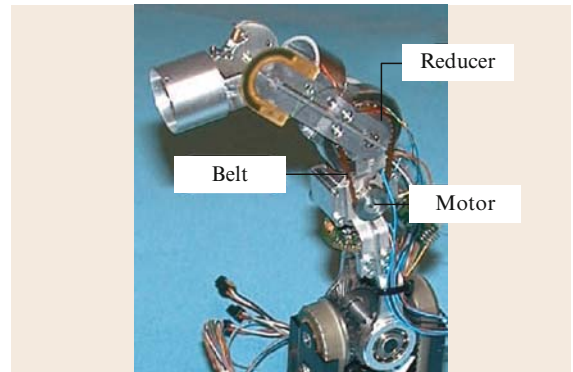


Fig. 15.15 Reducer location on the medial joint of the DLR hand II

architecture [15.30], for instance, in the design of the Utah/MIT dexterous hand (Fig. 15.17) in order to reduce static friction, the idea of using tendon sheaths was abandoned in favor of pulleys [15.3]. In order to find an optimal tradeoff between complexity and reliability of the mechanical arrangement and achieved friction level, a number of solutions that combine sheaths and pulleys for routing the tendon from actuators to fingers' joints have been adopted in the design of robot hands, e.g., the Stanford hand and the UB hand 3 reported in Fig. 15.18. The UB hand is characterized by an extremely simple structure, with the tendons completely routed within sheaths, but on the other hand the friction cannot be absolutely neglected and precise modeling of the interaction between the tendons and the tube is necessary for control purposes.

15.4.2 Transmission Model of Tendon-Outer-Tube System

This system can be modeled as shown in Fig. 15.19, where T_{in} , T_{out} , T_0 , ξ_{in} , ξ_{out} , R_i , x , and L are the tension at the input side, the tension at the output side, the initial pretension, the displacement at the input side, the displacement at the output side, the local radius of routing, the coordinate system along the wire, and the length of the tendon, respectively. The relationship between the tension at output and the input displacement is given by [15.31]

$$T_{out} - T_0 = K_t(\xi_{in}\phi_B), \quad (15.5)$$

where K_t and ϕ_B are the total stiffness and the equivalent backlash, respectively, given by

$$\frac{1}{K_t} = \frac{1}{K_e} + \frac{1}{K_s} + \frac{1}{K_{ap}}, \quad (15.6)$$

$$K_{ap} = K_w \frac{\lambda}{e^{\lambda} - 1}, \quad (15.7)$$

$$\phi_B = \frac{T_0 L}{EA} \cdot \frac{e^{\lambda} - \lambda - 1}{\lambda}, \quad (15.8)$$

$$\lambda = \sum |\beta_i| \mu \operatorname{sgn} \xi_{in}, \quad (15.9)$$

where K_e , K_s , K_w , K_{ap} , μ , E , A , and β_i are the stiffness of the environment, the stiffness of the force sensor, the apparent stiffness of the tendon, the friction coefficient, the Young's modulus, the cross-sectional area, and the bending angle of each segment of tendon, respectively. For example, $\sum |\beta_i| = 2\pi$ for the case given in Fig. 15.19. As can be seen from this example, the friction-related parameter λ increases dramatically when

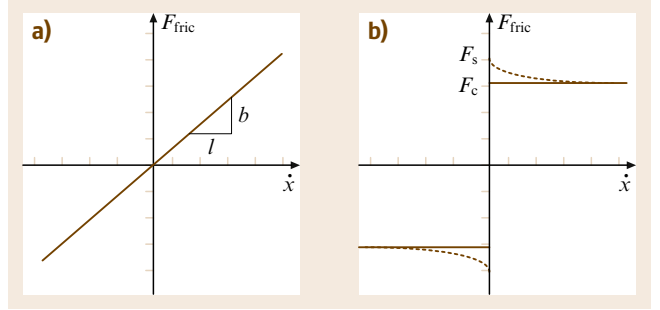


Fig. 15.16a,b Frictional phenomena: (a) viscous friction and (b) stiction and Coulomb friction

the tube is heavily bent. While we have the great advantage of choosing a free route for the power transmission, it results in a large nonlinearity for the transmission system. We should note that, while both the apparent stiffness of the tendon and the equivalent backlash vary with λ which is a function of the curvature of the route



Fig. 15.17 The Utah/MIT robotic hand

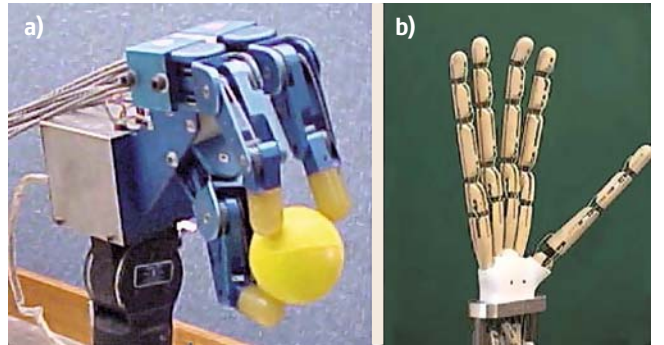


Fig. 15.18a,b Tendon-based robot hands: (a) Stanford/JPL hand and (b) UB hand 3

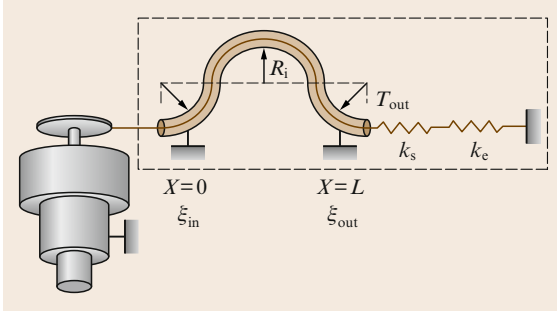


Fig. 15.19 Model of tendon-outer-tube transmission

as well as the friction coefficient, ϕ_B and K_{ap} result in $\phi_B = 0$ and $K_{ap} = K_w$ under $\mu = 0$. From the control viewpoint, such hysteresis is, of course, undesirable. To cope with these issues, each tendon should be designed to be as short as possible, so that we may keep high stiffness and small backlash in the transmission system.

15.4.3 Control Through Single-Acting Actuators

The use of single-acting actuators (i.e., standard motors with tendinous transmission), which are commonly assumed in the design of robot hands with remote actuation, requires the adoption of specific control techniques in order to guarantee the desired torques at the joints and to maintain at each time a positive tension on the tendons. To this purpose, the tendons are treated like inelastic frictionless elements and the problem is addressed in a way completely decoupled from the issue of device stability, discussed in previous sections.

A tendon, routed in the finger structure through sheaths or/and pulleys, can be modeled by means of an *extension function* $l_i(\theta)$ [15.32] that relates the joints' configuration with the tendon elongation. In the case of the tendon network represented in Fig. 15.2, the extension functions of the three tendons have the form:

$$l_i(\theta) = l_{0i} \pm R\theta_1 \pm R\theta_2,$$

where R is the radius of the pulleys and $\theta = [\theta_1 \ \theta_2]^T$ is the vector of the joint variables. Once the extension function has been determined, it is straightforward to derive the relationship between the tendons forces and the resulting joints torques. In fact, the relation between the joint speeds $\dot{\theta}$ and the tendon speeds \dot{l} can be deduced by simply differentiating the expression of extension functions

$$\dot{l} = \frac{\partial l}{\partial \theta}(\theta)\dot{\theta} = P(\theta)\dot{\theta}. \quad (15.10)$$

Because of the conservation of the power, from (15.10), one can deduce

$$\tau = P^T(\theta)f, \quad (15.11)$$

where the τ are the torques exerted on the joints, and the f are the force applied by tendons. From (15.11) it results that the force transmitted by a tendon may affect (and, in general, will affect) more than one joint.

In order to guarantee the possibility of exerting joint torques in every direction under the constraint of pure tensile forces, for any $\tau \in \mathbb{R}^n$ there must exist a set of forces $f_i \in \mathbb{R}^m$ such that

$$\tau = P^T(\theta)f \quad \text{and} \quad f_i > 0, i = 1, \dots, m, \quad (15.12)$$

where n and m are, respectively, the number of joints and the number of tendons. In this case the tendon network is said *force closure*. If the condition expressed by (15.12) is verified, given a desired torque vector τ it is possible to compute the force that the actuators must provide to the tendons according to

$$f = P^\dagger(\theta)\tau + f_N, \quad (15.13)$$

where $P^\dagger = P(P^T P)^{-1}$ is the pseudo-inverse of the coupling matrix P^T and $f_N \in \mathcal{N}(P^T)$ is a vector of *internal forces* that ensures that all tendon tensions are positive. In general, internal forces will be chosen as small as possible, so that the tendons are always taut but are not subject to excessive strains.

15.4.4 Control of a Robot Hand

The modeling and control aspects described in the previous sections, although very important and fundamental, can be considered as a sort of *low-level* problem in the control of a robot hand, in the sense that they are related to the specific physical properties of the device.

There are also other problems that must be faced and solved in order to operate in a profitable manner with a multifingered hand. These problems are solved by a proper design of the *high-level* control for the hand, which must take into account the interaction of the hand with the objects and more generally with the environment. In this context, general aspects that must be considered are: the control of forces/torques applied at the contact points, the necessity to model contact compliance/friction effects, the type of mobility both for the fingers and at the contact (rolling, sliding, etc.), a suitable planning algorithm for grasping and/or manipulating the objects, and so on.

These problems are illustrated in details in Chap. 26, Chap. 27, and Chap. 28.

15.5 Applications and Trends

In the industrial environment, simplicity and cost are the main guidelines for the design of end-effectors, and therefore simple devices, such as open/close grippers, are very common and widely used. This situation has led over the years to the development of a number of special-purpose devices, optimized for single specialized operations but not suitable for other tasks. At the moment, dexterous multifingered hands have not really been applied to any major application, mainly because of problems of reliability, complexity, cost.

On the other hand, more and more operations are currently envisaged for robots working in environments designed for, and utilized by, human operators. Entertainment, maintenance, space exploration, help to disable persons are just a few examples the of use of robotic systems in which interaction with tools and objects designed for human beings (or directly with them) is implied. In all these circumstances, the robot must be able to grasp and manipulate objects (different in dimension, shape, weight, etc.) similarly to humans, and therefore a robot hand, with a proper number of fingers and joints and also with an anthropomorphic appearance, seems to be the most adequate solution.



Fig. 15.20 The NASA/JPL Robonaut

At the moment, there are several projects aimed at the development of anthropomorphic robots. Among the most well known, one can mention the NASA/JPL Robonaut [15.18] (Fig. 15.20), the devices developed at the DLR, and the several projects on humanoid robots currently under development.

15.6 Conclusions and Further Reading

The design of multifingered robot hands has attracted the interest of the research community since the early days of robotics, not only as a challenging technical problem itself but probably also because of anthropomorphic motivations and the intrinsic interest in better knowledge of the human body. In the last decades, as previously discussed, several important projects have been launched, and important examples of robot hands have been developed. Nevertheless, the current situation is that reliable, flexible, dexterous hands are still not available for real applications. For these motivations, it is also easy to foresee in the future a consistent research activity in this fascinating field, with developments at the technological (sensor, actuator, material, etc.) and methodological (control, planning, etc.) levels.

Important connections with other scientific fields, such as cognitive science, are also expected.

As this research is so wide, it is not simple to suggest to interested readers further readings, except for classic books such as [15.32–34]. In fact, depending on the specific research area, many publications are available, although often not organized as reference books, mainly as technical papers published in journals or presented at international conferences. Moreover, since hundreds of new papers are published every year covering the different aspects of this robotic field, it is really quite difficult, and also not fair, to give specific suggestions for further readings at this time. We can therefore only refer to the citations already provided in the chapter.

References

- 15.1 T. Okada: Object-handling system for manual industry, *IEEE Trans. Syst. Man Cybern.* **2**, 79–86 (1979)
- 15.2 K.S. Salisbury, B. Roth: Kinematics and force analysis of articulated mechanical hands, *J. Mechan. Trans. Actu. Des.* **105**, 35–41 (1983)

- 15.3 S.C. Jacobsen, E.K. Iversen, D.F. Knutti, R.T. Lohnsan, K.B. Biggers: Design of the Utah/MIT dextrous hand, Proc. IEEE Int. Conf. Robot. Autom. ICRA86 (1986)
- 15.4 J. Butterfass, G. Hirzinger, S. Knoch, H. Liu: DLR's multisensory articulated hand. Part I: Hard- and software architecture, Proc. IEEE Int. Conf. Robot. Autom. ICRA99 (1999)
- 15.5 C. Melchiorri, G. Vassura: Mechanical and control features of the university of Bologna hand version 2, Proc. IEEE/RSJ Int. Conf. Int. Robot. Syst. IROS'92 (Raleigh 1992) pp.187–193
- 15.6 W.T. Townsend: MCB – industrial robot feature article– Barrett hand grasper, Ind. Robot. **27**(3), 181–188 (2000)
- 15.7 H. Kawasaki, T. Komatsu, K. Uchiyama: Dexterous anthropomorphic robot hand with distributed tactile sensor: Gifu hand II, IEEE/ASME Trans. Mechatron. **7**(3), 296–303 (2002)
- 15.8 T.J. Doll, H.J. Schneebeli: The Karlsruhe hand, Preprint IFAC Symp. Robot Contr. SYROCO (1988) pp.1–6
- 15.9 N. Fukaya, S. Toyama, T. Asfour, R. Dillmann: Design of the TUAT/Karlsruhe humanoid Hand, Intell. Robot. Syst. **3**, 1754–1759 (2000)
- 15.10 A. Bicchi, A. Marigo: Dexterous grippers: putting nonholonomy to work for fine manipulation, Int. J. Robot. Res. **21**(5–6), 427–442 (2002)
- 15.11 M.C. Carrozza, C. Suppo, F. Sebastiani, B. Massa, F. Vecchi, R. Lazzarini, M.R. Cutkosky, P. Dario: The SPRING hand: development of a self-adaptive prosthesis for restoring natural grasping, Auton. Robots **16**(2), 125–141 (2004)
- 15.12 J.L. Pons, E. Rocon, R. Ceres, D. Reynaerts, B. Saro, S. Levin, W. Van Moorleghe: The MANUS-hand dextrous robotics upper limb prosthesis: mechanical and manipulation aspects, Auton. Robots **16**(2), 143–163 (2004)
- 15.13 T. Iberall, C.L. MacKenzie: Opposition space and human prehension. In: *Dextrous Robot Hands*, ed. by T. Iberall, S.T. Venkataraman (Springer, New York 1990)
- 15.14 A. Bicchi: Hands for dextrous manipulation and robust grasping: a difficult road toward simplicity, IEEE Trans. Robot. Autom. **16**(6), 652–662 (2000)
- 15.15 M.R. Cutkosky: On grasp choice, grasp models, and the design of hands for manufacturing tasks, IEEE Trans. Robot. Autom. **5**(3), 269–279 (1989)
- 15.16 J. Butterfass, M. Grebenstein, H. Liu, G. Hirzinger: DLR-hand ii: next generation of a dextrous robot hand, Proc. IEEE Int. Conf. Robot. Autom. ICRA01 (Seoul 2001)
- 15.17 C. Melchiorri, G. Vassura: Mechanical and control features of the UB hand version II, Proc. IEEE/RSJ Int. Conf. Int. Robot. Syst. IROS'92 (1992)
- 15.18 R.O. Ambrose, H. Aldridge, R.S. Askew, R.R. Burridge, W. Bluethmann, M. Diftler, C. Lovchik, D. Magruder, F. Rehnmark: Robonaut: NASA's space humanoid, IEEE Int. Syst. (2000)
- 15.19 L. Birglen, C.M. Gosselin: Kinetostatic analysis of underactuated fingers, IEEE Trans. Robot. Autom. **20**(2), 211 (2004)
- 15.20 I. Yamano, T. Maeno: Five-fingered robot hand using ultrasonic motors and elastic elements, Proc. IEEE Int. Conf. Robot. Autom. (2005) pp.2684–2689
- 15.21 Shadow Dextrous Hand, (Shadow Robot Ltd., London 2007) <http://www.shadow.org.uk/>
- 15.22 M. Kaneko, M. Higashimori, R. Takenaka, A. Namiki, M. Ishikawa: The 100G capturing robot –too fast to see, Proc. 8th Int. Symp. Artif. Life Robot. (2003) pp.291–296
- 15.23 W. Paetsch, M. Kaneko: A three fingered multi-jointed gripper for experimental use, Proc. IEEE Int. Workshop Int. Robot. Syst. IROS'90 (1990) pp. 853–858
- 15.24 H. Maekawa, K. Yokoi, K. Tanie, M. Kaneko, N. Kimura, N. Imamura: Development of a three-fingerd robot hand with stiffness control capability, Mechatronics **2**(5), 483–494 (1992)
- 15.25 A. Bicchi: A Criterion for optimal design of multiaxis force sensors, J. Robot. Auton. Syst. **10**(4), 269–286 (1992)
- 15.26 A. Pugh: *Robot Sensors: Tactile and Non-Vision*, Vol. 2 (Springer, Berlin, Heidelberg 1986)
- 15.27 H.R. Nicholls, M.H. Lee: A survey of robot tactile sensing technology, Int. J. Robot. Res., **8**(3), 3–30 (1989)
- 15.28 W.T. Townsend, J.K. Salisbury: Mechanical bandwidth as a guideline to high-performance manipulator design, Proc. IEEE Int. Conf. Robot. Autom. ICRA89 (1989)
- 15.29 S.D. Eppinger, W.P. Seering: Three dynamic problems in robot force control, IEEE Trans. Robot. Autom. **8**(6), 751–758 (1992)
- 15.30 W.T. Townsend, J.K. Salisbury: The effect of coulomb friction and stiction on force control, Proc. IEEE Int. Conf. Robot. Autom. ICRA87 (1987)
- 15.31 M. Kaneko, T. Yamashita, K. Tanie: Basic considerations on transmission characteristics for tendon drive robots, Proc. 5th Int. Conf. Adv. Robot. (1991), 827–832
- 15.32 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)
- 15.33 J. Mason, J.K. Salisbury: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
- 15.34 M.R. Cutkosky: *Robotic Grasping and Fine Manipulation* (Springer, New York 1985)

Legged Robots

16. Legged Robots

Shuuji Kajita, Bernard Espiau

In this chapter, we introduce legged robots. After introducing the history of legged robot research in Sect. 16.1, we start to discuss hopping robots and analyze a simple passive walker as a typical cycling walking robot in Sect. 16.2; the Poincaré map is one of the most important tools to analyze its dynamics and stability. In Sect. 16.3, the dynamics and control of general biped robots are discussed. The key is the forward dynamics subject to the unilateral constraint between the feet and the ground. Its formal treatment leads to walking trajectory generation and various control methods. As a practical scheme to control biped robots, we discuss the zero-moment point (ZMP) in Sect. 16.4, including its definition, physical meaning, measurement, calculation, and usage. In Sect. 16.5, we move to multilegged robots. In this field, the most important subject is the relationship between gaits and stability. We also introduce the landmark robots in this field. In Sect. 16.6, we overview the divergence of the legged robots. We see leg-wheel hybrid robots, leg-arm hybrid robots, tethered walking robots, and wall-climbing robots. To compare these legged robots with different configurations, we use some useful performance indices such as the Froude number and the specific resistance, which are introduced in Sect. 16.7. We conclude the chapter and address future trends in Sect. 16.8.

16.1 A Brief History	362
16.2 Analysis of Cyclic Walking	363
16.2.1 A Few Points About Hopping Robots	363
16.2.2 Stability of Passive Walking	363
16.3 Control of Biped Robots Using Forward Dynamics	366
16.3.1 Configuration Space	366
16.3.2 Dynamics	366
16.3.3 Trajectory Generation	367
16.3.4 Control	368
16.4 Biped Robots in the ZMP Scheme	371
16.4.1 Mechanisms	371
16.4.2 Zero-Moment Point (ZMP)	371
16.4.3 Computed ZMP: ZMP Calculated from Robot Motion	373
16.4.4 ZMP-Based Walking Pattern Generation	375
16.4.5 ZMP-Based Walking Control	377
16.4.6 Expansion of the ZMP Concept	377
16.5 Multilegged Robots	378
16.5.1 Analysis of Static Gait	378
16.5.2 Practical Gait Design	380
16.5.3 Dynamic Quadrupeds Inspired by Mammals	382
16.5.4 Behavior-Based Multilegged Robots	382
16.6 Other Legged Robots	383
16.6.1 Leg-Wheel Hybrid Robots	383
16.6.2 Leg-Arm Hybrid Robots	383
16.6.3 Tethered Walking Robots	384
16.6.4 Wall-Climbing Robots	384
16.7 Performance Indices	385
16.7.1 Expansion of the Stability Margin Concept	385
16.7.2 Duty Factor and Froude Number	385
16.7.3 Specific Resistance	386
16.8 Conclusions and Future Trends	386
References	387

The idea of designing and realizing artificial entities is almost as old as humanity. After some wonderful formal studies by Leonardo Da Vinci, the first notable artificial machines, called *automatons* were built, among others

by Jacquard, Jacquet-Droz, Vaucanson (including his famous duck) mainly in France around the 18th century. This was followed by a florilegium of realizations from 1850 to World War I. The area then experienced a long

eclipse, up to the early 1970s. It should be noticed that the revival was then driven by scientists, instead of the

brilliant engineers, artists or magicians of the previous centuries.

16.1 A Brief History

The pioneering works in the field of legged robots were achieved around 1970 by two famous researchers, Kato and Vukobratovic. Both works were characterized by the design of relevant experimental systems. In Japan, the first anthropomorphic robot, WABOT 1, was demonstrated in 1973 by I. Kato and his team at Waseda University. Using a very simple control scheme, it was able to realize a few slow steps in static equilibrium. This achievement was the starting point of a prolific generation of legged robots in Japan.

In parallel, M. Vukobratovic and his team were very involved in the problems generated by functional rehabilitation. At the Mihailo Puppin Institute, Belgrade, Yugoslavia, they designed the first active exoskeletons, and several other devices such as the *Belgrade's hand*, but the most well-known outcome remains their analysis of locomotion stability, which exhibited around 1972 the concept of zero-moment point (ZMP), widely used since that time [16.3]. This was the first attempt to formalize the need for dynamical stability of legged robots; the idea was to use the dynamic wrench in order to extend a classical criterion of static balance (the center of mass should project inside the convex hull of contact points). This important point will be detailed later in this Chapter.

In the next decade, the breakthroughs came from the United States. Following the early work of R. McGhee in the 1960s at USC (University of Southern California), then in the 1970s at OSU (Ohio State University), which resulted in the first computer-controlled walking machine, M. Raibert started to study dynamically stable running at CMU (Carnegie Mellon University). Then, he launched the Massachusetts Institute of Technology (MIT) LegLab, where a sequence of active hopping robots, with one, two or four legs were designed, with impressive results, among them a famous flip performed by a two-legged hopping machine. Simultaneously, R. McGhee and K. Waldron, after the building of some prototypes, achieved the design of the largest hexapod in the world, called the *adaptive suspension vehicle*, a quasi-industrial system able to walk on natural irregular terrain, which was driven by a human (Sect. 16.5.2).

A third key period for research in legged robots was the early 1990s. Indeed, the idea of studying

purely passive mechanical systems was pioneered by *McGeer* [16.4]. In this seminal paper, McGeer introduces the concept of natural cyclic behavior, for a class of very simple systems: a plane compass on an inclined plane. Stable walking results from the balance between increase of the energy due to the slope and loss at the impacts. However, what should be emphasized here is that McGeer popularized for roboticists the analysis of such systems in terms of orbital stability using Poincaré maps. Several researchers have followed the tracks open by McGeer, with many extensions (examples in Fig. 16.1): adding trunk, feet and knees [16.1], semipassive control, walking/running underactuated systems like the Rabbit robot [16.2], etc.

Finally, the end of the millennium was a period of intense technological activities. Industrial breakthroughs showed to the world that building true humanoids was now possible. In Japan, the first humanoid robot, P2, was exhibited by Honda in 1996, followed by several more. Presently, the most impressive technical achieve-

a) Denise (2004)



b) RABBIT (2003)



Fig. 16.1a,b Underactuated biped robots [16.1, 2]

ments are still realized by industrial companies: ASIMO (Honda), **QRIO** (Sony), HRP (Kawada), being the major examples today, among others. In parallel, it should be noticed that the market for small humanoid robots, mainly aimed at entertainment, has grown steadily over the last decade.

While examining the history above and the present state of the art, it is clear that roboticists are now facing a challenge. Very nice technological achievements are available, especially biped robots. However, the ability of these systems to walk truly autonomously on uneven and various terrains in a robust way, i.e., in daily life, remains to be demonstrated. The goal of this chapter is therefore to provide some keys in modeling and recent

control advances, in order to be able to design adequate and efficient control schemes when needed. This will be based on two main classes of approaches: the use of so-called forward dynamics on one hand, and the use of the **ZMP** on the other hand. The Chapter is organized as follows: after a brief summary of the control principles used for hopping and passive robots, we will focus on the issues needed for the control of biped robots from a dynamic model scheme: modeling aspects, stability issues, trajectory generation, and control. Then we will present in depth the concept of **ZMP** and its use in a control scheme. Since the first parts are mainly concerned with biped robots, the chapter will end with a section dedicated to multilegged robots.

16.2 Analysis of Cyclic Walking

16.2.1 A Few Points About Hopping Robots

Cyclic legged robots are those that reach, either naturally or with the help from a control, a steady-state behavior characterized by a cycle in the phase plane. The underlying assumption is that there exists in some sense a more or less hidden set of optimal natural behaviors of the system. Within this class, hopping (or bouncing) robots are interesting since they are generally unstable, but capable of high performance in terms of velocities.

As mentioned, these robots have been widely studied in the MIT LegLab. It is not the goal of the Chapter to develop in depth the related design approaches and control techniques, and we refer the reader to the excel-

lent, although ancient, book by *Mark Raibert* [16.5] on the subject. Let us just give a flavor of Raibert's results, taken from this reference.

The basis of the work is in fact the planar one-leg hopping robot. Raibert proved that the control of such system could be split into three separate components: the first controls the altitude by providing a fixed thrust during each cycle; the second part controls the forward velocity of the whole system by assigning to the foot, at each step, a given distance from the hip when landing; and the last one controls the body attitude by servoing the hip during the stance phase. The related algorithms are quite simple and allow a real-time implementation. What is very interesting is that this simple approach applies almost straightforwardly to the case of the three-dimensional (3-D) one-legged hopping robot (Fig. 16.2b), and moreover this three-part control scheme can also be extended to biped (Fig. 16.2a) or quadruped robots by adding techniques of leg sequencing and using the concept of a virtual leg when pairs of legs operate in unison.

Indeed, this interesting piece of work was not really followed up, but was surely the inspiration for many researches on cyclic systems. Among these, purely passive walkers have largely been considered, and we will now provide some insight into this area.

16.2.2 Stability of Passive Walking

The aim of this section is to present some basic facts and concepts related to passive walking. Many more details may be found in the literature, for example

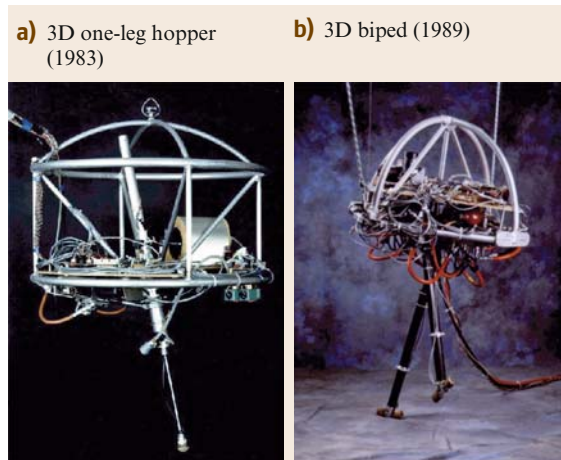


Fig. 16.2a,b Raibert's hopping robots [16.5]

in [16.1, 4, 6] among others. The issues considered here are mainly taken from [16.7]. We use the simplest possible model (Fig. 16.3), an unactuated symmetric planar compass descending a slope of angle ϕ . Masses are pin-point, and telescopic massless legs are only a way of ensuring foot clearance. Several assumptions underlie the model. Among these, let us mention that the swing phase is assumed to be slipless, and that the double stance phase, during which the swing and support leg are exchanged, is instantaneous. The related impact is slipless and inelastic. We define (following the notation in Fig. 16.3):

$$\begin{cases} \mu = \frac{m_H}{m}; \beta = \frac{b}{a} \\ \mathbf{x} = [\mathbf{q}, \dot{\mathbf{q}}]^\top = [q_{ns}, q_s, \dot{q}_{ns}, \dot{q}_s]^\top. \end{cases} \quad (16.1)$$

The swing-stage equations of the robot, similar to those for a frictionless double pendulum, can be written in the form of Lagrangian dynamics

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \frac{1}{a}\boldsymbol{\tau}_g(\mathbf{q}) = \mathbf{0}, \quad (16.2)$$

where

$$\begin{aligned} \mathbf{H}(\mathbf{q}) &= \begin{pmatrix} \beta^2 & -(1+\beta)\beta \cos 2\alpha \\ -(1+\beta)\beta \cos 2\alpha & (1+\beta)^2(\mu+1)+1 \end{pmatrix}, \\ \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) &= \begin{pmatrix} 0 \\ -(1+\beta)\beta \dot{q}_{ns} \sin(q_s - q_{ns}) \\ (1+\beta)\beta \dot{q}_s \sin(q_s - q_{ns}) \\ 0 \end{pmatrix}, \\ \boldsymbol{\tau}_g(\mathbf{q}) &= \begin{pmatrix} g\beta \sin q_{ns} \\ -[(\mu+1)(1+\beta)+1]g \sin q_s \end{pmatrix}, \end{aligned}$$

where 2α is the inter-leg hip angle.

The specificity of this system with respect to, for example, manipulation robots, is that we have to complete the continuous dynamics with equations describing the step transition. We will encounter this requirement of separate modeling in legged locomotion again when presenting the dynamics of bipedal walking (Sect. 16.3.2).

The pre- and the post-impact configurations of the robot can be related by $\mathbf{q}^+ = \mathbf{S}\mathbf{q}^-$, where \mathbf{S} is a 2×2 antisymmetric matrix with unit elements. The principle of conservation of angular momentum applied to the robot gives $\mathbf{Q}^-(\alpha)\dot{\mathbf{q}}^- = \mathbf{Q}^+(\alpha)\dot{\mathbf{q}}^+$, from

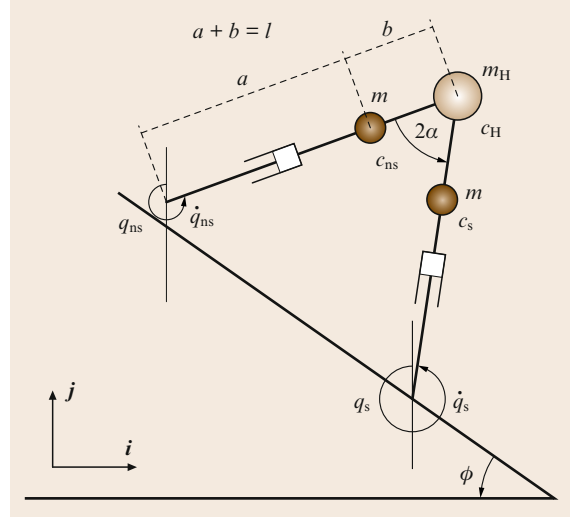


Fig. 16.3 Compass model

which we can obtain the joint velocity relationship $\dot{\mathbf{q}}^+ = [\mathbf{Q}^+(\alpha)]^{-1} \mathbf{Q}^-(\alpha) \dot{\mathbf{q}}^- = \mathbf{A}(\alpha) \dot{\mathbf{q}}^-$, where

$$\begin{aligned} \mathbf{Q}^-(\alpha) &= \begin{pmatrix} -\beta & -\beta + [\mu(1+\beta)^2 + 2(1+\beta)] \cos 2\alpha \\ 0 & -\beta \end{pmatrix}, \\ \mathbf{Q}^+(\alpha) &= \begin{pmatrix} \beta[\beta - (1+\beta) \cos 2\alpha] \\ \beta^2 \\ (1+\beta)[(1+\beta) - \beta \cos 2\alpha] \\ \cdots + 1 + \mu(1+\beta)^2 \\ -\beta(1+\beta) \cos 2\alpha \end{pmatrix}. \end{aligned}$$

The complete state vector \mathbf{x} before and after impact can thus be written as

$$\mathbf{x}^+ = \mathbf{W}(\alpha) \mathbf{x}^- \quad (16.3)$$

with matrix $\mathbf{W}(\alpha) = \begin{pmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}(\alpha) \end{pmatrix}$.

The periodic dynamic behavior of this system can be summarized in the phase portrait given in Fig. 16.4, where discontinuities result from impacts. The stability of this system can be analyzed in terms of *orbital stability*, a theoretical definition of which can be found in [16.7]. Intuitively, this means that, when the system deviates from its trajectory in the phase plane within a certain domain (the basin of attraction), its natural be-

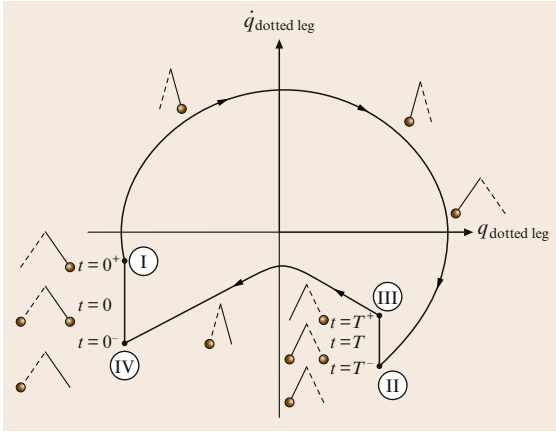


Fig. 16.4 Typical cycle

havior is to return to this phase-plane trajectory, called the limit cycle.

The concept of orbital stability is well suited to the analysis of cyclic systems such as steady-state walking. Thus, the robustness of the obtained gait can be assessed by measuring the size of the basin of attraction. However, for a general nonlinear system, the proof of the existence of a limit cycle, the analysis of its local orbital stability, and the procedure to compute the cycle and its basin of attraction are often difficult. For example, in the present case, the analysis would require the explicit integration of the dynamics during the swing phase. Nevertheless, it is possible to test the local stability of a limit cycle, once it has been found. One method to determine the stability of the robot gait is through the numerical computation of its Poincaré

map [16.4]. Limit cycles are fixed points of this map, which in the context of biped locomotion is called the *stride function* [16.4]. Essentially the procedure consists of injecting small perturbations to the robot states around the limit cycle and calculating the eigenvalues of the sensitivity matrix. For an orbitally stable cycle the eigenvalues lie within the unit circle. A natural choice of the Poincaré section corresponds to the state where the swing leg hits the ground. This introduces a natural discretization of the dynamics. For two successive touchdowns of the same leg, the states of the robot are related as:

$$\mathbf{x}_k = \mathbf{F}(\mathbf{x}_{k+1}). \quad (16.4)$$

The equilibrium is the solution of $\mathbf{x}^* = \mathbf{F}(\mathbf{x}^*)$. By taking a first-order expansion, we have

$$\mathbf{F}(\mathbf{x}^* + \Delta\mathbf{x}^*) \approx \mathbf{x}^* + (\nabla\mathbf{F})\Delta\mathbf{x}^*. \quad (16.5)$$

Therefore, $(\nabla\mathbf{F})\Delta\mathbf{x}^* \approx \mathbf{F}(\mathbf{x}^* + \Delta\mathbf{x}^*) - \mathbf{x}^*$, and a numerical procedure can be used to check whether the moduli of the eigenvalues of $(\nabla\mathbf{F})$ are strictly less than 1. One method is to apply successively smaller perturbations to every state and observe their first-return map, then to numerically solve the equation $(\nabla\mathbf{F})\boldsymbol{\tau} = \boldsymbol{\Psi}$, where the 4×4 diagonal matrix $\boldsymbol{\tau}$ collects the state variable perturbations and $\boldsymbol{\Psi}$ the measured variations in the first-return map. Note that this method is quite general and can be applied to other kind of mechanical systems that exhibit a periodic behavior.

It has been shown that passive walking robots can exhibit stable limit cycles. Often, for a given slope, ϕ , and for a particular range of slopes, the number of such stable behaviors is one. Interestingly, increasing ϕ may lead to the appearance of period doubling, evolving towards a chaotic behavior if ϕ is too large (Fig. 16.5).

In conclusion, it should be emphasized that this class of machines has inspired the design of efficient simple biped robots. Indeed, systems that exhibit natural passive gaits are *optimal* in some energetic sense. A limit cycle is analogous to the comfortable walking gait of humans, in which the consumption of metabolic energy per unit distance is minimal. An interesting idea is therefore to add to a passive system a minimum set of actuators in order to just compensate for the loss of energy when the system is not on a descending slope. Several laboratory realizations are based on this principle, but this idea has yet to be exploited for operational legged robots.

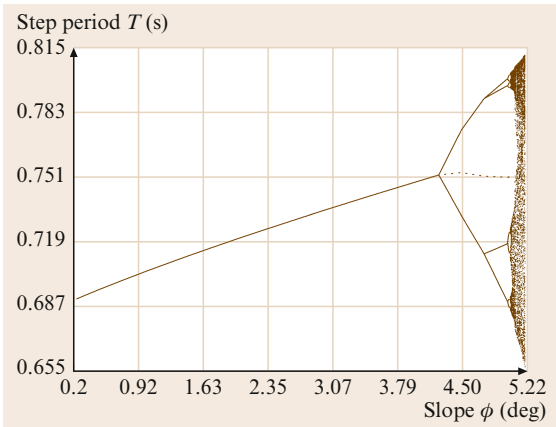


Fig. 16.5 Bifurcations and chaos

16.3 Control of Biped Robots Using Forward Dynamics

This section presents a view of the state of the art in modeling and control of biped robots in the framework of forward dynamics. Space does not allow to go deeply into the details of the equations or their derivation, which is why we only provide the reader with the essential issues and refer to more complete papers, such as [16.8, 9] and mainly the deep analysis of [16.10, 11].

16.3.1 Configuration Space

A biped robot is usually modeled as a rigid tree-form tridimensional articulated structure. Basically, the internal parameterization is q_1 , while that of the joint space is Q . If the robot is considered as having its foot (resp. feet) motionless during every phase of swing (resp. stance), then the model is that of a classical robot manipulator. If all joints are actuated, then the closure equations make the robot overactuated during the stance phase (with two feet on the ground). However, it is more correct and general to consider a walking robot as a free system in space, but subject to nonconstant unilateral contacts. The configuration space is therefore $\{\{Q\} \oplus R^3 \oplus SO(3)\}$, where the six-dimensional (6-D) displacement of a given body is parameterized by q_2 .

16.3.2 Dynamics

A basic modeling approach consists of splitting the dynamics equations into three parts: Lagrangian dynamics in the full space; constraints due to contact forces; and transition equations with impacts.

Continuous Dynamics and Contact Forces

Under the assumption that the robot structure is rigid, continuous dynamics can be expressed in the Lagrangian form

$$H(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \Gamma + \Gamma_{\text{ext}}, \quad (16.6)$$

where $q = (q_1, q_2) \in \mathcal{R}^n$ is defined in Sect. 16.3.1. $\Gamma = [0, \tau]^\top \in \mathcal{R}^n$ is the generalized efforts vector, including the joint actuation torques (generally bounded), $\tau \in \mathcal{R}^m$; H is the inertia matrix, C is the matrix of centrifugal, gyroscopic, and Coriolis effects, and τ_g is the generalized gravity force vector. The set (q, \dot{q}) constitutes the *state* of the robot.

The points of the robot in contact with the ground satisfy a closure equation of the form

$$\phi(q) = \begin{pmatrix} \phi_n(q) \\ \phi_t(q) \end{pmatrix} = 0,$$

where Γ_{ext} are the torques generated by these ground contact, which can be expressed as

$$\Gamma_{\text{ext}} = J(q)^\top \lambda(q, \dot{q}),$$

where $J(q) = \frac{\partial \phi(q)}{\partial q}^\top$ is the Jacobian matrix of the contact points of the robot, i. e., where the external forces are applied. $\lambda(q, \dot{q})$ are Lagrange multipliers. We can split this expression into two parts

$$\Gamma_{\text{ext}} = J_n(q)^\top \lambda_n(q, \dot{q}) + J_t(q)^\top \lambda_t(q, \dot{q}).$$

The subscripts ‘n’ and ‘t’ stand for normal and tangential to the ground components, respectively. The Lagrange multipliers λ_n and λ_t express the amplitudes of those efforts.

Now, there are two main ways of modeling the ground interaction. First, it is possible to consider a viscoelastic model, but this suffers from possible problems in the physical analysis and numerical integration. We will consider here the opposite case: that the contacts are rigid and result in a set of constraints, called unilateral, such as nonpenetration, to which should be added a friction model. The semipositive normal forces and accelerations of the contact points are therefore related by a complementarity condition [16.10, 12]

$$\lambda_n^\top(q, \dot{q}) \ddot{\phi}_n(q) = 0, \lambda_n(q, \dot{q}) \geq 0, \ddot{\phi}_n(q) \geq 0.$$

Excluding the case where the system is slipping, tangential constraints can be written as

$$\ddot{\phi}_t(q) = 0.$$

Finally, since the existence of friction limits the allowed tangential forces, a nonslip condition is satisfied as long as

$$\|\lambda_t\| \leq \mu_0 \lambda_n,$$

where μ_0 is the friction coefficient, which depends on the materials in contact.

Impacts

As stated in Sect. 16.2.2, if the impact is assumed to be inelastic and nonsliding, the velocity of the system jumps from \dot{q}^- to \dot{q}^+ according to the equations

$$H(q)(\dot{q}^+ - \dot{q}^-) = J_n(q)^\top \Lambda_n(q, \dot{q}) + J_t(q)^\top \Lambda_t(q, \dot{q}),$$

where $\Lambda(q, \dot{q}) = [\Lambda_n(q, \dot{q}), \Lambda_t(q, \dot{q})]^\top$ is the vector of the impulse forces.

Under the assumption that contact points do not move after impact, we can write:

$$\dot{\phi}_n(q) = J_n(q)\dot{q}^+ = 0 \quad \text{and} \quad \dot{\phi}_t(q) = J_t(q)\dot{q}^+ = 0$$

Dynamic Balance

Basically, a biped robot is in equilibrium if it maintains a walk without falling. In the early times of robotics, the gaits were *statical*. This means that, when the velocities and accelerations are small, and assuming that contact points are motionless during each phase of the step, the balance condition reduces to the classical one: the projection of the robot's center of mass belongs to the convex hull of contact points. Therefore, a static walk can be defined as a continuous sequence of configurations ensuring the forward progression and maintaining the erected position of the system simultaneously such that at each instant the static equilibrium condition is satisfied. As we will see later, related control schemes are easy to design.

However, the true interesting issue is dynamic balance. Nicely addressed in the case of Raibert's hopping robots or for passive walkers through orbital stability as presented in Sect. 16.2.2, the questions of stability and balance deserve a deeper analysis in the case of other kinds of biped robots, such as the work described in [16.11].

For a biped robot, *the dynamical equilibrium can be intuitively linked to the idea of a possible movement*. Let us firstly return to the dynamics; splitting the equations in a way coherent with the two parts of the configuration space, the dynamics (16.6) can be rewritten as:

$$\begin{cases} H_1(q)\ddot{q} + C_1(q, \dot{q})\dot{q} = \tau + J_1(q)^\top \lambda - \tau_{g1}(q), \\ H_2(q)\ddot{q} + C_2(q, \dot{q})\dot{q} = 0 + J_2(q)^\top \lambda - \tau_{g2}(q), \end{cases} \quad (16.7)$$

where τ is the set of actuator torques.

It can be shown [16.11] that the left-hand side of the second equation of (16.7) is equivalent to the dynamic wrench of the system, while the right-hand side is equivalent to the wrench of contact and gravity forces. This equation is of the Newton–Euler form, where the Newton part can be expressed easily in terms on the acceleration of the center of mass of the robot. This shows a fundamental issue: the global displacement and orientation of the robot can only be realized through contact forces. Moreover, this motion is necessarily associated with a change of posture. More generally, the system can achieve a desired movement if and only if the total

wrench of gravity and contact forces is equal to the dynamic wrench of the robot. For a given control scheme, the question is then to check whether this property is satisfied by the robot under control.

In the case where all contact points are in the same plane, it can be shown that there exists a point in this plane around which the horizontal rotation momentum of gravity and dynamic forces vanishes. This point is known as the *zero-moment point (ZMP)*, and is also the *center of pressure (CoP)*. We will see in Sect. 16.4 how this concept allows the derivation of efficient control schemes.

Another way of setting the problem of the dynamic balance of a walking machine while also taking into account nonhorizontality and tangential forces is to state the problem as follows. A necessary condition for a walking system to realize a motion specified by a trajectory $q(t)$ on a given time interval is that there exists contact forces $\lambda(t)$ such that

$$\begin{cases} H_2(q)\ddot{q} + C_2(q, \dot{q})\dot{q} + \tau_{g2}(q) = J_2(q)^\top \lambda, \\ A(\lambda) \geq 0, \end{cases} \quad (16.8)$$

where the vector inequality $A(\lambda) \geq 0$ gathers all constraints on normal (unilaterality) and tangential (Coulomb friction) forces. Finally, the motion can definitively be achieved if the actuation forces are compatible with the dynamical requirements of the first equation of (16.7). As an illustration of this, we will propose in the next section a control method in relation to this approach.

16.3.3 Trajectory Generation

A frequently used approach for designing a full biped control scheme is to combine the a priori definition of desired trajectories to follow, a classical control allowing them to be tracked, and dedicated online adaption techniques to cope with model uncertainties, obstacles, and disturbances to prevent the robot from falling. Two main classes of methods can be used to compute the desired (also called *reference*) trajectories: they can either be obtained from capturing human motions, or be purely computer generated. In the latter case, the usual approach includes two steps: first, a set of output variables with adequate dimension is chosen. This is generally made from the three-dimensional (3-D) coordinates of a few selected points on the robot. Secondly, after parameterization (under the form of splines, for example) the desired trajectories of these variables are computed for every phase of the gait: stance, swing, left or right, etc.,

allowing one to take into account, for example, obstacles or stairs. They can be either all synchronized in time, or be functions of a single timed reference coordinate. The most interesting way of achieving this computation is to use an optimization algorithm based on the dynamics that takes into account various constraints. The trajectories either remain specified in the output space or are transformed in trajectories in the joint space, $q_d(t)$. Finally, these trajectories are tracked by a controller, as explained later; an online walk stabilization mechanism is often added.

Another approach to be mentioned is the possibility to use bio-inspired techniques, such as the central pattern generator. The idea is to design self-oscillating systems (i.e., without inputs, although the shape of the output is tunable through some parameters) from which synchronized periodic motions of the joints can be derived. This approach is generally used for generating gaits for multilegged robots (quadrupeds, hexapods) or snake-like systems, but some works also address biped robots [16.13]. The most well-known nonlinear oscillator is the Van der Pol equation

$$\ddot{y} + a(1 - by^2)\dot{y} + c^2y = 0, \quad (16.9)$$

from which many variations can be derived. Similar results can be obtained from a more biologically inspired point of view, through the design of neural oscillators [16.14]. In this case, sets of artificial neural networks with possible open-loop sinusoidal excitation are connected to generate plausible walking patterns. In some cases it is even possible to create a kind of feedback to cope with environmental variations or to improve stability [16.15].

16.3.4 Control

We present in this subsection three examples of control schemes only, without details. Indeed, many kinds of control schemes are reported in the literature and it is not possible in a single chapter to provide an exhaustive view. We therefore refer the reader to the best journals and conferences in this field for more information [16.16–23].

Simple Dynamic Control

For simplicity, let us only consider the case where the desired walking behavior is specified as a set of multidimensional reference joint trajectories, $q_d(t)$, defined for every distinct walking phase. It is now necessary to design a control scheme allowing these trajectories to be tracked as accurately as possible in the ideal case. Basi-

cally, a proportional–(integral)–derivative (PID) loop on the tracking error is the core of such a control scheme. However, since the use of high gains in a PID controller is not always desirable, due both to the presence of noise and to the need for discretization, good tracking performance require the integration of mechanical modeling issues into the control. When the dynamics of the system are well known, it is often interesting to use them, in a so-called computed torque approach, in which the applied control torque is of the generic form

$$\begin{aligned} \Gamma = & \hat{H}(q)(k_p(q - q_d) + k_v(\dot{q} - \dot{q}_d) + \ddot{q}_d) \\ & + \hat{C}(q, \dot{q})\dot{q} + \hat{\tau}_g(q) + \hat{I}_F, \end{aligned} \quad (16.10)$$

where, in relation to the dynamics of (16.6), the hats mean that approximate models are used. \hat{I}_F covers friction aspects and the k_* are control gains. This form is given in the joint space, but can be also used in other diffeomorphic output spaces. Concerning the models, let us note that, at least, the compensation of gravity term $\hat{\tau}_g(q)$ should be considered. Moreover, one of the most important source of errors is the effect of internal friction. As soon as an effective model of friction in joints, gears, and actuators, \hat{I}_F , is available, it should be used, of course with care. In practice, if foot switches exist, phase transitions, leg synchronization, and reinitialization of the trajectories on their cycle can be triggered by the events generated by such sensors (see also Sect. 7.4)

This kind of control can be used at least for generating static walking. If the trajectories have been generated in a dynamical framework, for example, by taking into account a desired position of the ZMP, it can also be used in dynamic walk, possibly with the addition of a dedicated stabilization algorithm. Many variations of this scheme exist based, for example, the linearization of the dynamics, or the use of an inverted-pendulum model.

Nevertheless, although this kind of control can overcome some internal disturbances, it is neither sufficient to ensure the stability of *walking* in a robust enough way in real time, nor able to cope with environmental uncertainties. The basic requirement is now to find a way of controlling the ground contact forces in order to be sure that the support foot (or feet) remain motionless when it is specified and/or needed by the reference trajectory. This means that the control has to ensure that the inequality conditions (no sliding, no take-off), which involve the Lagrange multipliers as stated in Sect. 16.3.2, are verified at each time despite disturbances. Nevertheless, this capacity of reaction will be limited by the bounds that exist on the actuators.

An adequate way of addressing the question of online walk stabilization is to share the control in two parts [16.24, 25]: a first part, which is devoted to the tracking of the prescribed trajectory and involves a large subset of the joints and a second part, dedicated to the control of the ground forces through selected joints (the trunk, or more often, the ankles). It can effectively be shown that controlling ankle torques is a way of approximately control ground reaction forces through an adequate compliant model. Like the similar scheme called *hybrid control* in robot manipulators (see Sect. 7.4), the implementation can use the contact wrenches more or less directly, estimated from force/torque sensors integrated within the ankles or the soles of the feet.

A Variation: The Use of Parameterized Trajectories

A drawback of the classical tracking trajectory approach is that it is necessary to explicitly design as many trajectories as possible situations. Therefore, adaptation to disturbances requires a jump from one trajectory to another, for example, in order to stride over a small obstacle or when it is required to increase or lower the overall velocity by modifying the shape, duration, and length of the prescribed step. An interesting attempt to derive a systematic way of online adaption of the behavior of the system consists of adding degrees of freedom to the definition of the trajectories to be tracked by parameterizing them [16.26]. The principle of the method is described below.

Let us consider that, in fact, the reference trajectory depends on a set of time-varying twice-differentiable parameters p : $q_d = q_d[p(t)]$. Various characterizations of the trajectories may appear in p : step length, maximal height of the heel trajectory, time scaling, etc. These parameters are reflected, for example, in the coefficients of the splines that define the trajectories. They are set to a nominal value p^* corresponding to the desired motion. However, since they are supposed to evolve when needed, it is necessary to set their dynamics of return to p^* , through a linear second-order behavior $\ddot{p}_d = f(p, \dot{p}, p^*)$. Now, it can be shown that it is possible to gather the dynamics, including a PD control scheme like (16.10), the unilaterality of contacts, the requirements of nonsliding and no take-off, and the bounds on actuators, all described in Sect. 16.3.2 and summarized in (16.8), into a single vector inequality of the form

$$A(q, \dot{q}, p, \dot{p}, \ddot{p}) \leq 0. \quad (16.11)$$

The problem is now solved by finding at each time through an adequate optimization method (such as FSQP (feasible sequential quadratic programming), originally developed at ISR (Institute of Systems Research), University of Maryland) the parameter acceleration \ddot{p} that minimizes $\|\ddot{p}_d - \ddot{p}\|^2$ while ensuring that (16.11) is satisfied. This approach is, for example, able to compensate for disturbances such as external forces applied to the body, since they are reflected at the internal-state level, and are therefore accounted for through the respect of the inequality (16.11). It has been shown [16.26] that it is possible to increase the range of acceptable disturbances by a factor ten using this method, compared to nonadaptive approaches. Furthermore, the use of exteroceptive sensors such as distance, proximity, and vision is allowed as soon as they directly generate a modification of the parameters, for example, in order to climb a stair or avoid an obstacle.

Online Optimization

The ultimate way to adapt the motion of the robot in real time is to even avoid using any type of precomputed trajectories. This requires that both the desired motion and the related control can be computed online. In the most general case, one can even imagine that these two steps could be fused into a single approach. Besides, thanks to the exponential growth of available embedded computing power, it is now possible to envision the computation of accurate dynamical models and/or the use of complex optimization techniques in real time. This is why we propose to end this part of the chapter with the description of promising optimization-based methods that avoid the explicit synthesis of analytical trajectories.

Model Predictive Control (MPC). The idea is to start from well-known *model predictive control* (MPC) techniques based on the following principle. Let us assume that a dynamical model of the system that is good enough for synthesizing a control scheme is available. Then, the following operations are achieved at each sampling time:

1. measurement of the actual state
2. computation of the control that optimizes a given state-dependent cost function on a finite horizon, starting from the current discrete time
3. application of the control input at the first time index only
4. return to step 1

A main drawback of the standard approach is that the available theoretical results of stability are presently

limited to the linear case or to very particular classes of nonlinear systems. For example, the nonlinear model predictive control (NMPC) presents stability properties under certain assumptions. Moreover, it has some ability to handle constraints, which makes it well suited to the problem of walking pattern synthesis and control of a biped robot subject to unilateral constraints or disturbances due to an unstructured environment. A complete overview of theoretical and practical results concerning the MPC or the NMPC can be found in [16.27]. However, it can be seen from these papers that applying such algorithms is not straightforward when dealing with highly dynamical systems such as robots.

Trajectory-Free NMPC. Following the idea of the parameter adaptation method presented earlier based on the real-time minimization of a cost function through an SQP (sequential quadratic programming) method, an alternative approach consists of solving at every time point a constrained optimization problem with a moving horizon [16.28]. A set of constraints is imposed in order both to ensure the feasibility of walking and to satisfy the user's requirements. Defining these constraints is then the only way which is used to express all the issues involved in the motion of the robot.

As an example of such specifications, let us consider the swing phase in normal steady-state walking. The *inequality* constraints belong to two distinct subsets.

1. Intrinsic constraints related to safety issues:
 - respecting physical limits: the admissible control torques belong to a given set, which may depend on the instantaneous joint velocities; the range of angular joint positions is bounded;
 - ensuring stability: all along the swing phase, the normal contact forces at the support foot are

strictly greater than given positive values; the absolute values of the tangential forces are less than given thresholds that depend on the friction parameters.

2. The constraints associated with the gait specification:
 - achieving forward progression: the horizontal velocity of the ankle of the swing leg belongs to a given positive interval; the horizontal position of the pelvis stays inside the position of the toes of the support foot and the horizontal position of the heel of the swinging foot;
 - controlling the posture: the angle expressing the trunk bending is positive and bounded; the angle between the sole of the swing foot and the ground is fixed; the height of the pelvis is low-bounded;
 - ensuring foot clearance: the altitude of the ankle is constrained to stay inside a given area (for example, specified by two polynomial functions of its horizontal position). These functions are a way of avoiding obstacles or climbing stairs.

All these constraints can be gathered in the single vector inequality $g(.) \leq 0$. The *equality* constraints are mainly:

- the dynamics itself (16.6) instanced for the swing phase
- the initial conditions on the state: at time t_{k+1} , they are the final state at time t_k

Therefore the method consists of optimizing, from the inputs (control, state, contact forces, etc.), the anticipated future behavior of the system, subject to the constraints above, using an internal model over a finite sliding time horizon, which is the prediction horizon. Since the problem is finally termed open-loop constrained optimization, solved at each sampling time, the last step is to define an objective function to minimize. Usually, such a cost function includes a term linked to the final state over the prediction horizon and a quadratic expression reflecting some energetic considerations.

The solution of the optimization algorithm is a sequence of N_c control inputs over the prediction horizon N_p . Only the first input is applied to the system and the procedure starts again. It should be noted that the feedback effect, which should be included in any real-time control, appears through the use of *actual* current values of the state in the optimization. Furthermore, the adaptation of the motion in order to react to unexpected events can be performed by modifying online the set of equality and inequality constraints.

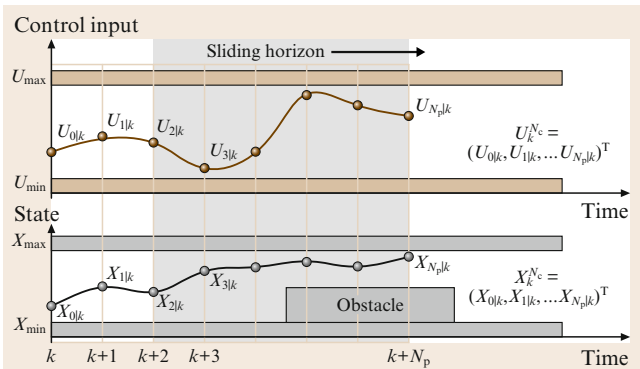


Fig. 16.6 NMPC principles

Let us consider Fig. 16.6, for example. Here, the state $X_{i|k}$ is the prediction of the heel altitude at time $k + i$, computed from the current time k . Then the detection of an obstacle leads to modification of the polynomial that is the lower limit of the trajectory of X . The prediction horizon finally allows one to anticipate the dynamical

behavior of the system, which ensures that the foot safely crosses over the obstacle.

This method is known as trajectory-free nonlinear model predictive control and is describe in detail in [16.28]. An extension of the predictive control method to the case of ZMP-based control is presented in [16.29].

16.4 Biped Robots in the ZMP Scheme

The zero-moment point (ZMP) might be one of the most famous technical terms born in robotics community. Figure 16.7 shows two important figures in the scene of ZMP-based biped walking. Figure 16.7a is WL-10RD, developed by Takanishi and Kato. This is the first ZMP-based robot, which successfully realized dynamic biped walking in 1985 [16.30]. It is a 12-degree-of-freedom (DOF) biped, 1.43 m high and weighing 84.5 kg, and driven by hydraulic actuators.

Figure 16.7b is ASIMO, a 26-DOF humanoid robot developed by Honda Motor Co. in 2000 [16.31]. This is one of the most famous robots in public culture, and at the same time, its superior performance of biped locomotion (walking and running) is well acknowledged by specialists. According to the published papers and patents, ZMP takes an important role in the walking control of ASIMO.

In this section, we describe the basic definition, the calculation, and the usage of ZMP.

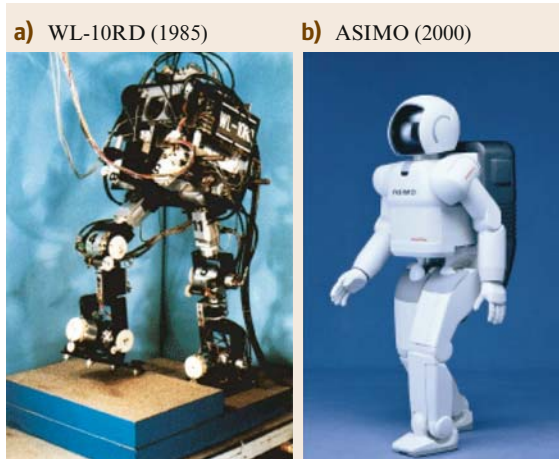


Fig. 16.7a,b Biped robots controlled in the ZMP scheme [16.30,31]

16.4.1 Mechanisms

Figure 16.8 shows recently developed biped robots controlled by the ZMP scheme. Figure 16.8a is Johnnie, developed by Gienger et al. in 2001 [16.32]. It is a 1.80 m-high 17-DOF humanoid weighing 40 kg, driven by direct-current (DC) servo motors with harmonic drive gears and ball screws. Figure 16.8b is HRP-2L, which was developed by Kaneko et al. [16.33]. It is a 1.41 m-high 12-DOF biped weighing 58.2 kg, driven by DC servo motors with harmonic drive reduction gears. Figure 16.8c is WL-16R, developed by Takanishi et al. [16.34] as a *walking chair* that can carry a human weighing up to 94 kg. It is a 1.29 m-high 12-DOF biped weighing 55 kg with Stewart-platform-type legs driven by electric linear actuators. Figure 16.8d is HUBO, developed by Oh et al. [16.35]. It is a 1.25 m-high 41-DOF humanoid robot weighing 55 kg.

Although these robots have different leg mechanism and outlook, they share some common features:

1. there are at least six fully actuated joints for each leg,
2. the joints are position controlled,
3. the feet are equipped with force sensors, which are used to measure the ZMP.

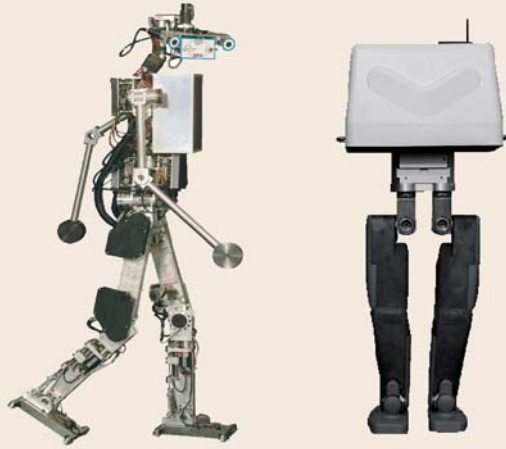
As we will see in the following subsections, these are the fundamental requirements for ZMP-based walking robots.

16.4.2 Zero-Moment Point (ZMP)

The term *zero-moment point* (ZMP) was coined by Vukobratović and Stepanenko in 1972. They said [16.36]:

In Fig. 16.9 an example of force distribution across the foot is given. As the load has the same sign all over the surface, it can be reduced to the resultant force R , the point of attack of which will be

a) Johnnie (2000) b) HRP-2L (2001)



c) WL-16R (2003) d) HUBO (2006)



Fig. 16.8a–d Biped robots controlled in the ZMP scheme [16.32–35]

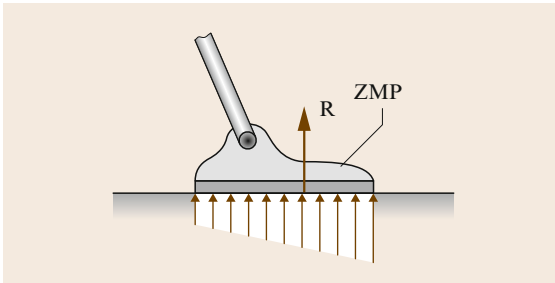


Fig. 16.9 Original definition of the zero-moment point (ZMP) [16.36]

in the boundaries of the foot. Let the point on the surface of the foot, where the resultant R passed,

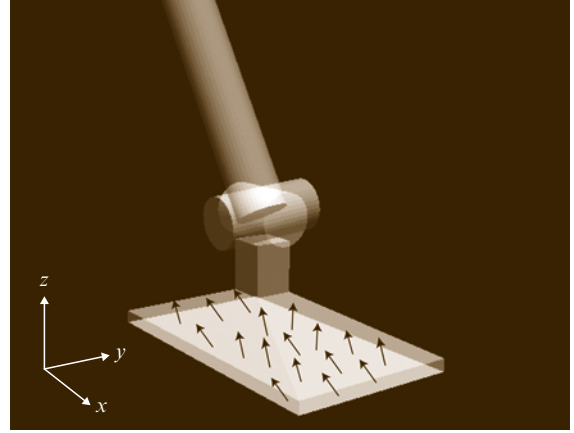


Fig. 16.10 Floor reaction force in three dimensions

*be denoted as the zero-moment point, or **ZMP** in short.*

We can see that the **ZMP** is defined as a center of pressure (CoP) of the floor reaction force.

For the further discussion, let us discuss detailed floor reaction force in three dimensions, as shown in Fig. 16.10.

Suppose that the floor reaction force is acting on a finite number of contact points $p_i (i = 1, \dots, N)$ and that each force vector is of the form

$$f_i := [f_{ix} f_{iy} f_{iz}]^T,$$

where f_{ix} , f_{iy} , and f_{iz} are the force components in the x , y , and z directions in the coordinate system fixed to the ground, respectively. The **ZMP** can be calculated as

$$p := \frac{\sum_{i=1}^N p_i f_{iz}}{\sum_{i=1}^N f_{iz}}. \quad (16.12)$$

It can also be written as

$$p = \sum_{i=1}^N \alpha_i p_i, \quad (16.13)$$

$$\alpha_i := f_{iz} / f_z, \quad (16.14)$$

$$f_z := \sum_{i=1}^N f_{iz}. \quad (16.15)$$

Since an ordinary walking robot cannot generate adhesive force on its soles,

$$f_{iz} \geq 0 \quad (i = 1, \dots, N). \quad (16.16)$$

Then we can say,

$$\begin{cases} \alpha_j \geq 0 & (i = 1, \dots, N), \\ \sum_{i=1}^N \alpha_i = 1. \end{cases} \quad (16.17)$$

The points that satisfy (16.13) and (16.17) form the *support polygon*, the convex hull of the contacting point. Thus, we can conclude that the **ZMP** always exists in the support polygon. In other words, *the ZMP never leaves the support polygon* as the result of the unilateral constraint on the floor reaction force.

Now let us calculate the torque around the **ZMP**:

$$\tau = \sum_{i=1}^N (\mathbf{p}_i - \mathbf{p}) \times \mathbf{f}_i. \quad (16.18)$$

This equation can be rewritten in terms of vector components as

$$\tau_x = \sum_{i=1}^N (p_{iy} - p_y) f_{iz} - \sum_{i=1}^N (p_{iz} - p_z) f_{iy}, \quad (16.19)$$

$$\tau_y = \sum_{i=1}^N (p_{iz} - p_z) f_{ix} - \sum_{i=1}^N (p_{ix} - p_x) f_{iz}, \quad (16.20)$$

$$\tau_z = \sum_{i=1}^N (p_{ix} - p_x) f_{iy} - \sum_{i=1}^N (p_{iy} - p_y) f_{ix}, \quad (16.21)$$

where p_{ix} , p_{iy} , p_{iz} are the components of the position vector \mathbf{p}_i and p_x , p_y , and p_z are the components of the **ZMP**.

When the floor is horizontal, we have $p_{iz} = p_z$ for all i , thus, the second term of (16.19) and the first term of (16.20) become zero. Moreover, by substituting (16.12) into (16.19) and (16.20), we obtain

$$\tau_x = \tau_y = 0. \quad (16.22)$$

This is the reason why \mathbf{p} was named as *zero-moment* point. Nevertheless, one must note that the frictional force creates a nonzero vertical moment (16.21) in the general case.

$$\tau_z \neq 0. \quad (16.23)$$

16.4.3 Computed ZMP: ZMP Calculated from Robot Motion

With given robot dynamics and motion, we can calculate or predict the resulting **ZMP** by using Newton's law. To distinguish this from the original **ZMP** defined in the

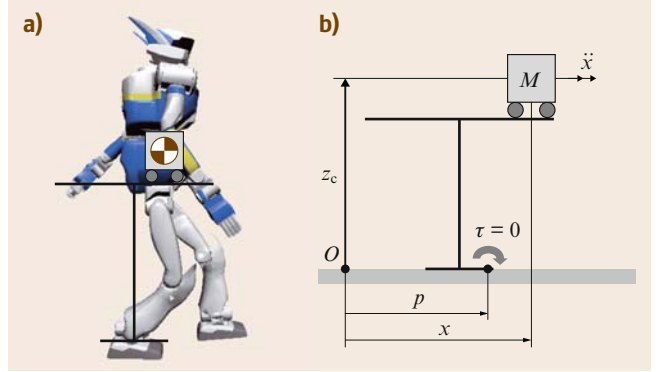


Fig. 16.11a,b Cart-table model [16.37]

former subsection, we will use the term *computed ZMP* as in the paper of Vukobratović et al. [16.3]

Simple Case

Let us start with an extremely simple mechanism. Figure 16.11a illustrates a walking robot and its simplified model, which consists of a running cart on a massless table. The cart has mass M and its position is (x, z_c) corresponds to the center of mass of the robot (Fig. 16.11b). Also, the table is assumed to have the same support polygon as the robot.

In this case, the torque τ around the point p is given by

$$\tau = -Mg(x - p) + M\ddot{x}z_c, \quad (16.24)$$

where g is the acceleration due to gravity. Using the zero-moment condition of $\tau = 0$, the computed **ZMP** of this cart-table model is obtained as

$$p = x - \frac{z_c}{g} \ddot{x}. \quad (16.25)$$

From this equation, we can observe two fundamental facts about the computed **ZMP**.

1. When the acceleration of the cart is zero, the **ZMP** corresponds to the projection of the CoM: $p = x$.
2. The computed **ZMP** is not bounded by the support polygon. In fact, we can easily determine x and \ddot{x} to give any specified value of p .

The second fact is explained in Fig. 16.12a. When the cart acceleration is too large, the computed **ZMP** goes outside of the support polygon. This happens since (16.25) does not consider the support polygon and the unilateral constraint (16.16). In other words, (16.25) assumes that the foot is glued to the floor. If the unilateral constraint is properly considered, we have the state of

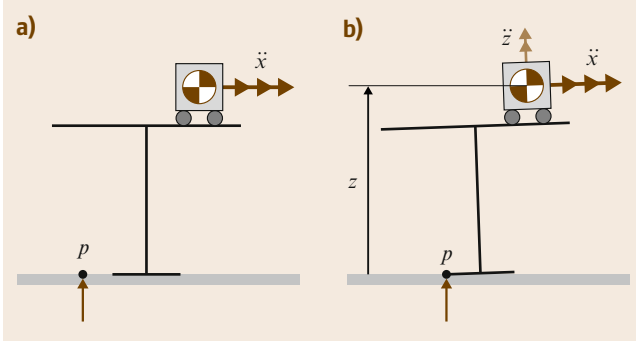


Fig. 16.12a,b Computed ZMP (a) fictitious case (b) correct result

Fig. 16.12b. Since the table is no longer upright, the computed ZMP must be calculated by

$$p = x - \frac{z}{g + \ddot{z}} \ddot{x}, \quad (16.26)$$

which gives the computed ZMP on the edge of support polygon.

In much of the literature, states like Fig. 16.12b are not explicitly discussed, but the computed ZMP is allowed to leave the support polygon under the implicit assumption of the glued foot on the floor. In this case, a computed ZMP outside of the support polygon implies that the robot might not maintain its feet in full contact with the ground and that the walking motion will not be performed as planned.

When the computed ZMP is inside the support polygon, it guarantees the full foot-ground contact under the unilateral constraint, while it tells us nothing about *stability* in the context of control theory.

Computed ZMP for Full 3-D Dynamics

We will show a method to calculate the computed ZMP of a robot that consists of N rigid-body links in three dimensions. As in the preceding treatment, we assume that the all kinematic information (the position of the CoM, link orientation, link velocity, etc.) have already been calculated by forward kinematics. In this subsection, the link postures and their angular velocities are represented in the ground-fixed coordinate system.

First we calculate the total mass M and the center of mass of the whole robot c by

$$M = \sum_{j=1}^N m_j, \quad (16.27)$$

$$c = \sum_{j=1}^N m_j c_j / M, \quad (16.28)$$

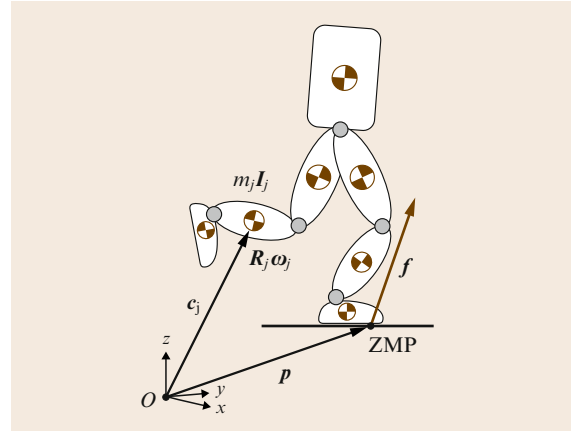


Fig. 16.13 Robot model and ZMP in three dimensions

where m_j and c_j are the mass and the CoM of the j -th link, respectively.

The total linear momentum \mathcal{P} is given as

$$\mathcal{P} = \sum_{j=1}^N m_j \dot{c}_j. \quad (16.29)$$

Then the total angular momentum \mathcal{L} with respect to the origin is

$$\mathcal{L} = \sum_{j=1}^N [c_j \times (m_j \dot{c}_j) + R_j I_j R_j^\top \omega_j], \quad (16.30)$$

where R_j , I_j , and ω_j are the 3×3 rotation matrix, inertia tensor, and angular velocity of the j -th link, respectively. $R_j I_j R_j^\top$ gives the inertia tensor in the ground-fixed frame.

Applied with the external force f and external moment τ , the change of linear and angular momenta are described by Newton and Euler's law:

$$f = \dot{\mathcal{P}} - Mg, \quad (16.31)$$

$$\tau = \dot{\mathcal{L}} - c \times Mg, \quad (16.32)$$

where $g := [00 - g]^\top$ is the vector of acceleration due to gravity.

Suppose that the external force is acting on the ZMP located at p .

$$\tau = p \times f + \tau_{\text{ZMP}}, \quad (16.33)$$

where τ_{ZMP} is the moment at the ZMP, whose first and second components are zero.

Substituting (16.31) and (16.32) into (16.33) we obtain

$$\tau_{\text{ZMP}} = \dot{\mathcal{L}} - c \times Mg + (\dot{\mathcal{P}} - Mg) \times p. \quad (16.34)$$

The first and the second rows of this equation are

$$\tau_{ZMP,x} = \dot{\mathcal{L}}_x + Mgy + \dot{\mathcal{P}}_y p_z - (\dot{\mathcal{P}}_z + Mg)p_y, \quad (16.35)$$

$$\tau_{ZMP,y} = \dot{\mathcal{L}}_y - Mgx - \dot{\mathcal{P}}_x p_z + (\dot{\mathcal{P}}_z + Mg)p_x, \quad (16.36)$$

where we use the notations

$$\begin{aligned} \tau_{ZMP} &= [\tau_{ZMP,x} \tau_{ZMP,y} \tau_{ZMP,z}]^T, \\ \mathcal{P} &= [\mathcal{P}_x \mathcal{P}_y \mathcal{P}_z]^T, \\ \mathcal{L} &= [\mathcal{L}_x \mathcal{L}_y \mathcal{L}_z]^T, \\ \mathbf{c} &= [x \ y \ z]^T. \end{aligned}$$

The zero-moment point is calculated from (16.35) and (16.36) using $\tau_{ZMP,x} = \tau_{ZMP,y} = 0$

$$p_x = \frac{Mgx + p_z \dot{\mathcal{P}}_x - \dot{\mathcal{L}}_y}{Mg + \dot{\mathcal{P}}_z}, \quad (16.37)$$

$$p_y = \frac{Mgy + p_z \dot{\mathcal{P}}_y + \dot{\mathcal{L}}_x}{Mg + \dot{\mathcal{P}}_z}, \quad (16.38)$$

where p_z is the height of the floor.

When the robot remains stationary, we have the **ZMP** as the projection of the CoM:

$$p_x = x, \quad (16.39)$$

$$p_y = y. \quad (16.40)$$

Note that **ZMP** can also be calculated from the motion equation in Lagrangian form. See Sect. 16.3.2 for more details.

16.4.4 ZMP-Based Walking Pattern Generation

Prescribing Foot and ZMP Trajectories

In the original work, a model with a compensating mechanism was assumed to realize the prescribed **ZMP** pattern [16.36]. Figure 16.14a illustrates an example model, which uses predetermined leg trajectories while **ZMP** is controlled by using the compensating mass. This concept was realized by the WL-12 robot developed by *Takanishi and Kato* (Fig. 16.14b) [16.38].

In modern implementations, the body motion is used to realize prescribed **ZMP** trajectories, as shown in Fig. 16.15. Instead of the leg joint trajectories, the foot trajectories are prescribed. The leg motion is determined by inverse kinematics from the body and the feet trajectories.

Figure 16.16 shows an example **ZMP** trajectory for two forward steps. First, the time profile of the support

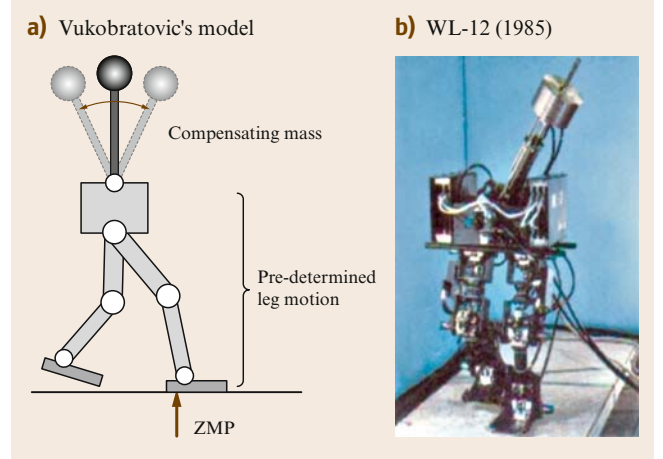


Fig. 16.14a,b Biped robots with compensating mass [16.38]

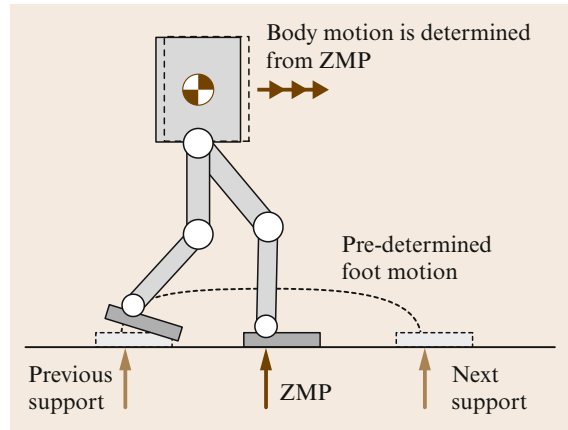


Fig. 16.15 Modern implementation of the **ZMP** scheme

polygon is determined from the foot trajectory as the gray band. Its width is determined from the foot geometry and the step length and changes at every touchdown (td) and lift-off (lo). Then the **ZMP** trajectory (bold line) is determined so that it runs within the support polygon. Note that we can design any **ZMP** trajectories as long as it remains inside of the support polygon with a certain stability margin.

Pattern Generation for the Desired ZMP

Let us define a function **ZMP()** that gives a calculated **ZMP** for the given robot motion.

$$\begin{pmatrix} p_x \\ p_y \end{pmatrix} = \mathbf{ZMP}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}), \quad (16.41)$$

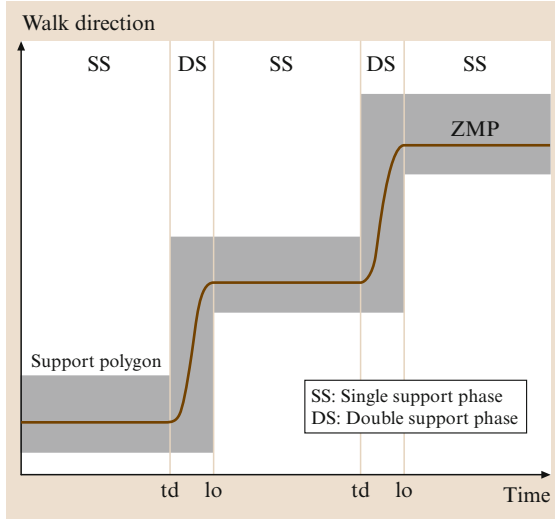


Fig. 16.16 Support polygon and prescribed **ZMP** trajectory. td: touchdown; lo: lift-off

where \mathbf{q} , $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ are the vectors of joint position, velocity, and acceleration, respectively. **ZMP**-based walking pattern generation is to determine the feasible $\mathbf{q}(t)$ that satisfies given desired **ZMP** $p_x^d(t)$, $p_y^d(t)$. It should be noted that:

1. As the premise of this method, the robot must be fully actuated and position controlled. Basically, underactuated or torque-controlled robots do not fit in this scheme.
2. When the dimension of \mathbf{q} is larger than three, extra constraints must be introduced to solve the redundancy. For example, constraints of constant body height and upright posture are frequently used.
3. Even under the proper constraints, (16.41) have infinite numbers of solutions because the initial condition $\mathbf{q}(0)$ is left free. This degree of freedom is used to avoid the divergence of the joint trajectories.
4. In practice, we do not need an exact control of **ZMP**: $p_x(t) = p_x^d(t)$, $p_y(t) = p_y^d(t)$. Nevertheless, it is preferred to minimize the sum of **ZMP** error during the walking motion.

Several practical methods have been proposed. *Takanishi* et al. proposed to solve this problem by using Fourier transformation [16.39]. By applying the fast Fourier transformation (FFT) to the **ZMP** reference, the **ZMP** equations can be solved in the frequency domain, then the inverse FFT returns the resulted CoM trajectory to the time domain. *Kagami* et al. proposed a method to

solve this problem in the discrete time domain [16.40]. They showed that the **ZMP** equation can be discretized as a trinomial expression, and that it can be solved efficiently by an algorithm of $O(N)$ for given reference data of size N .

Another practical method was described by *Huang* et al. [16.41]. *Sugihara* et al. proposed a method that can consider the multibody dynamics of a robot [16.42]. *Nagasaka* et al. proposed an efficient real-time method that is also applicable to running and jumping motion [16.43]. *Harada* et al. proposed another efficient real-time method that can be used when pushing an object during walking [16.44].

Pattern Generation Using Preview Controller

In this subsection, we will describe the method proposed by *Kajita* et al. [16.37]. Its stability and possible expansions are well discussed by *Wieber* [16.29].

For the simplicity, let us use the cart-table model of Fig. 16.11 again, but this time, we take the jerk of the cart as the system input u ,

$$\ddot{x} = u. \quad (16.42)$$

In this way, the **ZMP** equation (16.25) can be translated into a strictly proper dynamical system as

$$\frac{d}{dt} \begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u, \quad (16.43)$$

$$p = \begin{pmatrix} 1 & 0 & -z_c/g \end{pmatrix} \begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix}.$$

For this system, we can design a digital controller that lets the system output follow the reference input as

$$u(k) = -G_i \sum_{i=0}^k e(i) - G_x \mathbf{x}(k), \quad (16.44)$$

$$e(i) := p(i) - p^d(i),$$

where G_i is the gain for the **ZMP** tracking error, G_x is the state feedback gain, and $\mathbf{x} := [x \dot{x} \ddot{x}]^T$. The value at the k -th sample time is indicated by appending (k) . The block diagram of the feedback system is shown in Fig. 16.17.

Although we can guarantee the stability of this system by LQ (linear quadratic) feedback gains, the controller of (16.44) cannot realize sufficient **ZMP** tracking due to the phase delay. To solve this problem, we must

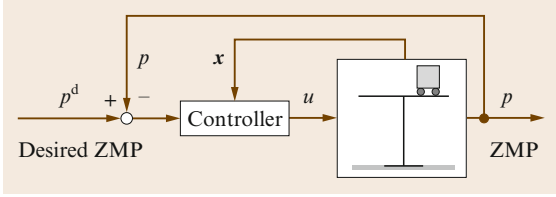


Fig. 16.17 ZMP tracking control

use the following controller:

$$u(k) = -G_i \sum_{i=0}^k e(i) - G_x x(k) - \sum_{j=1}^{N_L} G_p(j) p^d(k+j) \quad (16.45)$$

The third term is new and contains the ZMP reference up to N_L samples future. Since this controller uses future information, it is called a *preview controller* [16.45, 46]. The gain $G_p(j)$ is called the preview gain and Fig. 16.18 shows its profile towards the future. We can observe that the magnitude of the preview gain quickly diminishes, thus the ZMP reference in the far future can be neglected.

Figure 16.19 shows an example of a walking pattern generated by using a preview controller. The upper graph shows the sagittal motion along the x -axis and the lower graph shows the lateral motion along the y -axis. We can see that a smooth trajectory of the CoM (dashed line) is generated and that the resulting ZMP (bold line) follows the reference (thin line) with good accuracy.

16.4.5 ZMP-Based Walking Control

If we have an ideal mechanism on a perfect flat floor, we can expect that a robot walks just by a playback of prescribed joint trajectory. Since this is not the

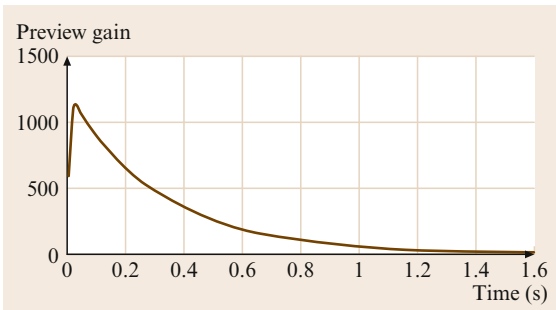


Fig. 16.18 Preview controller gain G_p (sample time $T = 5$ ms, $z_c = 0.814$ m, $Q_e = 1.0$, $Q_x = 0$, $R = 1.0 \times 10^{-6}$)

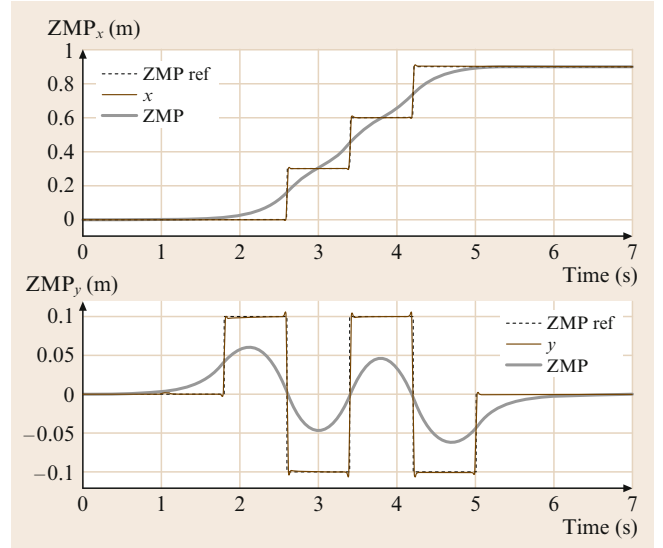


Fig. 16.19 Body trajectory obtained by preview control; previewing period $TN_L = 1.6$ s

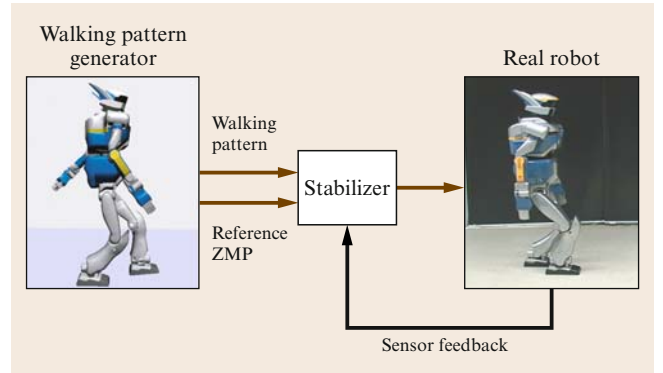


Fig. 16.20 Pattern generator and stabilizer

case in general, we need feedback control to modify the reference trajectory using sensor information, as shown in Fig. 16.20. Such stabilizers are discussed in [16.25, 47–50].

16.4.6 Expansion of the ZMP Concept

We can predict a robot fall when the CoM projection is observed outside of the support polygon. However, we cannot use the ZMP by this way, since the ZMP never leaves the support polygon of a real biped robot. When the ZMP is on the edge of the support polygon, all we can say is that *the robot can fall*, since it is merely a necessary condition of falling. As an alterna-

tive of **ZMP**, *Goswami* proposed a foot-rotating indicator (**FRI**) [16.51]. **FRI** is a **ZMP** calculated by omitting the acceleration of the supporting foot and is a measurable value of real robots that may leave the support polygon. When **FRI** is outside of the support polygon, this indicates a support foot acceleration, which breaks a foot–floor contact.

Another problem is that the **ZMP** is defined only on the flat plane. *Sardain* and *Bessonnet* discussed a general

ZMP calculation including a double support phase on uneven terrain [16.52].

Saida et al. proposed a feasible solution of wrench (**FSW**) as a new criterion for a legged robot on uneven terrain [16.53].

Hirukawa et al. also proposed a universal stability criterion that can treat a general geometry of foot–ground contact and unidirectional force constraints [16.54].

16.5 Multilegged Robots

This section treats robots that have more than three legs.

16.5.1 Analysis of Static Gait

Compared with bipeds, multilegged robots have a wider choice of foot placing to maintain static balance. From this reason, many research works have concentrated on gait planning for statically stable walking rather than treating dynamic stability. In this subsection, we will introduce those important results by following *McGhee* [16.55] and *Song* and *Waldron* [16.56].

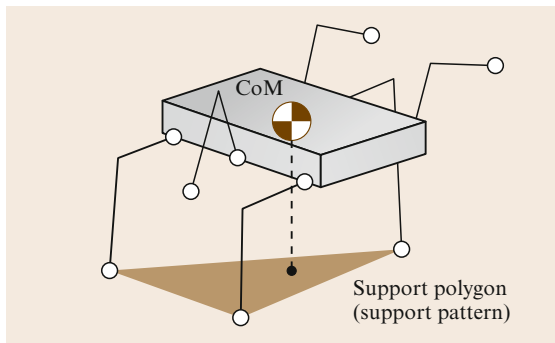


Fig. 16.21 Support polygon (support pattern) of a multilegged robot

Stability Margins

In multilegged robot research, the term *support pattern* is frequently used for support polygon [16.55–57]. By neglecting the inertial effects caused by body and leg acceleration, we can guarantee that the robot maintains balance if the projection of the CoM falls inside the support pattern, as shown in Fig. 16.21.

For a given configuration of a walking robot, the *stability margin* S_m is defined as the minimum distance of the vertical projection of CoM to the boundaries of the support pattern in the horizontal plane, as illustrated in Fig. 16.22a. In addition, an alternative index was proposed to obtain the optimal gait analytically. That is the *longitudinal stability margin* S_l , which is defined as the minimum distance from the vertical projection of the CoM to the support pattern boundaries along the line parallel to the body motion.

Quadruped Creeping Gait and Crawl Gait

From the front to the rear of a $2n$ -legged robot or animal, let us index the left legs and the right legs by odd numbers $1, 3, \dots, 2n-1$ and even numbers $2, 4, \dots, 2n$, respectively. Following this rule, the legs of a quadruped robot are numbered as in Fig. 16.23.

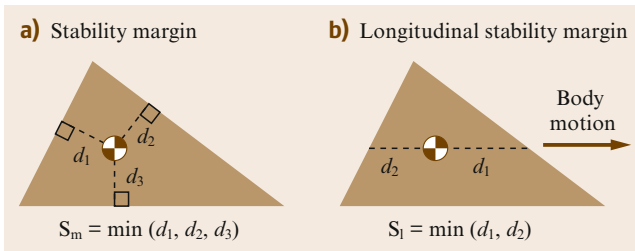


Fig. 16.22a,b Definition of stability margins

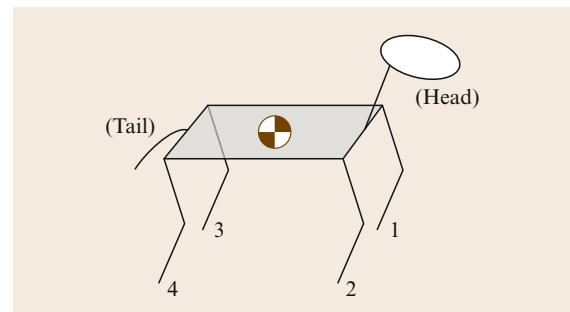


Fig. 16.23 Leg labeling of quadruped robot

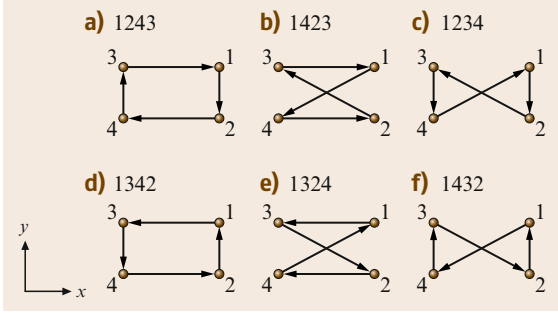


Fig. 16.24a–f Quadruped creeping gaits [16.57]

To walk keeping static stability, a quadruped robot must lift and place only one leg at each step. In general, such a pattern is called *creeping gait*. The possible creeping gaits of a quadruped robot can be expressed by a series of leg numbers to show the order of foot placing. Always choosing leg 1 as the first swing leg, we can distinguish $(4 - 1)! = 6$ different gaits, as shown in Fig. 16.24.

As we will see later, the 1423 creeping gait (Fig. 16.24b) gives the maximum stability for wading along the x -direction, and is called the *crawl gait*. Note that if the walking direction was $-x$, 1324 (Fig. 16.24e) gives the crawl gait. Likewise, 1234 (Fig. 16.24c) and 1432 (Fig. 16.24f) represent the crawl gait in the $-y$ and y directions, respectively.

On the other hand, the 1243 and 1342 creeping gaits (Fig. 16.24a, d) gives medium stability and are suitable for turning.

Gait Diagram

Figure 16.25b is a *gait diagram* to describe the gait sequence of a multilegged robot. The horizontal axis indicates the time normalized by walk cycle time T . The line segment associated with each leg starts from the touchdown and ends at lift-off. Therefore, the length of the line segment indicates the period of the support phase. From this diagram, we can define the *duty factor* β_i and *phase* ϕ_i of leg i as

$$\beta_i = \frac{\text{support period of leg } i}{T}, \quad (16.46)$$

$$\phi_i = \frac{\text{touchdown time of leg } i}{T}. \quad (16.47)$$

The touchdown time of leg i is measured from the touchdown of leg 1; therefore, we have $\phi_1 = 0$ for any gait.

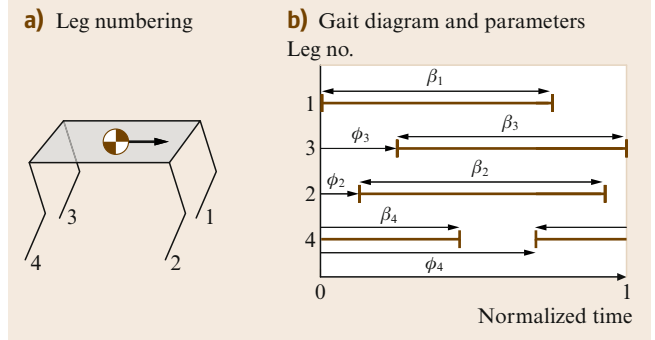


Fig. 16.25a,b Gait diagram and parameters

Wave Gait of Quadrupeds

There exists a special gait which gives maximum longitudinal stability margin for a quadruped robot [16.57]. It is called the *wave gait* defined as

$$\beta_i = \beta, \quad (i = 1, \dots, 4), \quad (16.48)$$

$$0.75 \leq \beta < 1, \quad (16.49)$$

$$\phi_2 = 0.5, \quad (16.50)$$

$$\phi_3 = \beta, \quad (16.51)$$

$$\phi_4 = \phi_3 - 0.5, \quad (16.52)$$

where β is the duty factor of the wave gait. Figure 16.26 shows the gait diagram of a wave gait of $\beta = 0.75$.

If we look at the order of foot placement in Fig. 16.26, it is a crawl gait (1423 creeping gait) shown in Fig. 16.24b. Therefore, a wave gait is the optimal crawl gait.

The most important feature of the wave gait is specified by (16.51). This means that leg 3 touchdowns at the moment of leg 1's lift-off, as indicated by the ellipse (broken line) in Fig. 16.26. Equation (16.49) gives the possible range of duty factor for a statically stable walking. In addition, a wave gait is characterized as *regular* and *symmetric*. A gait is regular when all legs have the same duty factor β , as specified by (16.48). A gait is

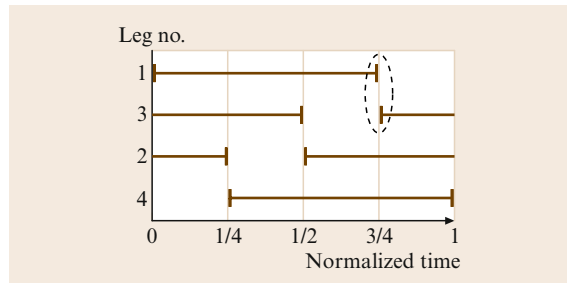


Fig. 16.26 Wave gait of a quadruped robot ($\beta = 0.75$)

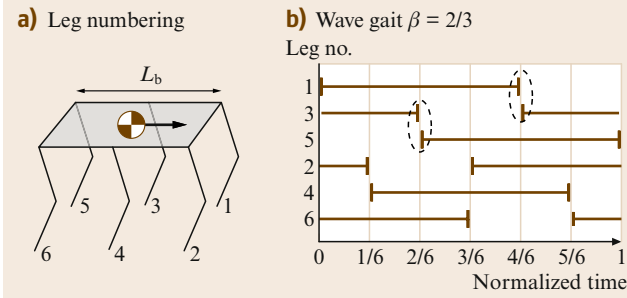


Fig. 16.27a,b Wave gait of a hexapod robot

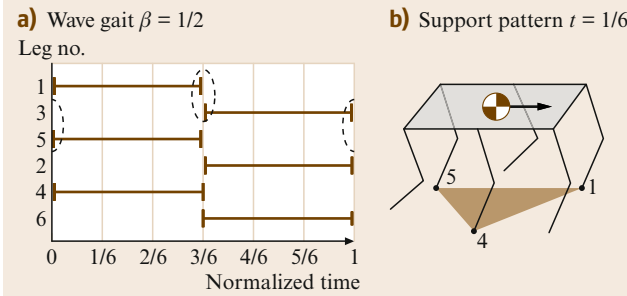


Fig. 16.28a,b Tripod gait of a hexapod robot

symmetric when the right and the left leg of each column has a phase difference of a half-cycle, 0.5 as specified by (16.50) and (16.52).

Wave Gait of $2n$ -Legged Robots

A wave gait of $2n$ -legged robots can be defined as a regular and symmetric gait with the following feature.

$$\phi_{2m+1} = F(m\beta), \quad (m = 1, \dots, n-1), \quad (16.53)$$

$$3/(2n) \leq \beta < 1, \quad (16.54)$$

where $F(x)$ gives the fractional part of a real number x . Equation (16.53) is a generalization of (16.51). An example wave gait for hexapod robot is shown in Fig. 16.27. The ellipses (broken line) show the condition (16.53).

Figure 16.28 shows a wave gait with $\beta = 1/2$, which is the most important for hexapod robots. From the constraint (16.54), this is the smallest duty factor for hexapods and thus results in the fastest walking speed. This gait is specially called a *tripod gait* since a robot is supported by the three legs 145 and 236 reciprocally (Fig. 16.28b).

It is known that the longitudinal gait stability margin of a $2n$ -legged robot is maximized by wave gaits [16.56]. Figure 16.29 shows the optimal wave-gait stability margin normalized by body length L_b (Fig. 16.27a) for

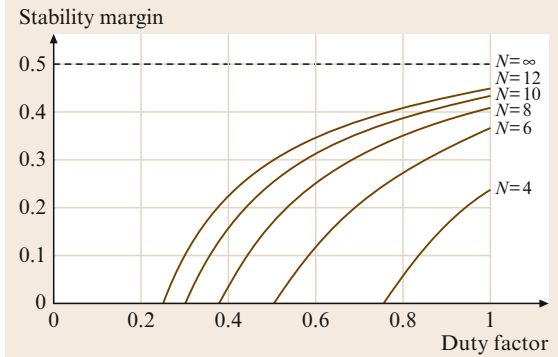
Fig. 16.29 Optimal wave-gait stability margin of N -legged locomotion systems [16.55]

Fig. 16.30 The adaptive suspension vehicle (1986) [16.58]

N -legged robots [16.55]. We can observe that by increasing the number of legs the stability and allowable range of duty factors are improved. The largest improvement is obtained from $N = 4$ to 6 and the gains gradually become small towards $N = \infty$. Since the cost of the hardware goes up in proportion to the number of the legs, this explains the rarity of developed robots with more than ten legs.

16.5.2 Practical Gait Design

Unlike the ideal analysis in previous section, real multi-legged robots are far more complex. In this subsection, we see more practical aspects of gait control by concentrating on a few state-of-the-art examples.

Hexapod: The Adaptive Suspension Vehicle

Figure 16.30 shows one of the most famous hexapods, the adaptive suspension vehicle (ASV) developed by Waldron and McGhee [16.56, 58]. The ASV is a hy-

Table 16.1 Specifications of the ASV [16.56]

Dimensions:	5.2 m in length
	3.0 m in height
	2.4 m in track width
Degree of freedom	18 DOF (3 DOF × 6)
Weight	2700 kg
Payload	220 kg
Speed	8 km/h
Grade climb ability:	60% gradient
Mobility:	Ditch crossing: 1.8 m
	Vertical step crossing: 1.7 m
	Isolated wall crossing: 1.4 m
	Fording depth: 1.2 m

Table 16.2 Gait used by the ASV [16.56]

Periodic	Wave gait
	Equal phase gait
	Backward wave gait
	Backward equal-phase gait
	Dexterous periodic gait
	Continuous follow-the-leader gait
Nonperiodic	Discontinuous follow-the-leader gait
	Large-obstacle gait
	Precision footing gait
	Free gait

draulically driven hexapod robot that can carry a parson over rough terrain. Table 16.1 lists the design specifications and mobility of the ASV at experimentation in natural terrain.

Table 16.2 shows the variety of gaits that was used to control the ASV. We can see that the wave gait appears at the top of list, however, it requires undesirable peak power consumption at certain moments. The *equal-phase gait* was designed to solve this problem. Another interesting gait is the *follow-the-leader gait* in which the middle foot is placed on the footprint made by the front foot, and the rear foot is placed in the footprint of the middle foot. By using this gait, the load on the human operator to specify foot placement when moving over difficult terrain can be drastically reduced. In this manner, each gait listed in Table 16.2 has some advantage depending on the ground adaptivity, power consumption or smoothness of body motion.

Quadruped: TITAN Series

Figure 16.31 shows quadruped robots developed by Hirose and his colleagues [16.59, 60]. TITAN III (Fig. 16.31a) has four legs, each 1.2 m long, and weighs

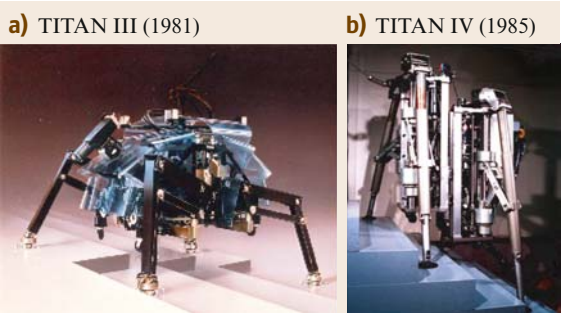


Fig. 16.31a,b The TITAN series [16.59, 60]

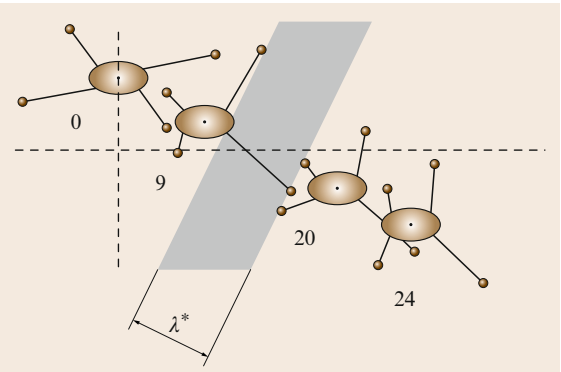


Fig. 16.32 Walking motion over a river [16.61]

80 kg. Each leg is driven by three DC motors, thus this is a 12-DOF walking robot. On flat ground, TITAN III uses a generalized crawl gait which allows the robot to move omnidirectionally (*crab-walk gait*). When the robot step into a rough terrain it uses *free gait*, which is a nonperiodic gait to search for safe footholds, avoiding obstacles and maintaining static stability. Figure 16.32 shows an example of free gait to walk over a river. Once the robot returns to flat ground, it is controlled to return the crab-walk gait, which is more efficient.

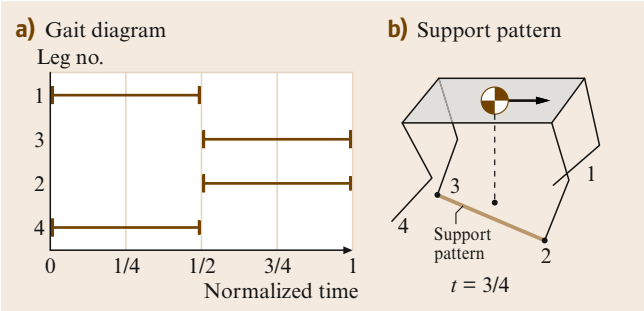


Fig. 16.33a,b Trot gait

TITAN IV (Fig. 16.31b) has a similar configuration to TITAN III, but weighs 160 kg. Since it has much powerful motors, an experiment of dynamic walk was conducted. For a given return speed of the swing leg U , the walking speed with duty factor β is

$$V = \frac{1-\beta}{\beta} U. \quad (16.55)$$

For a statically stable wave gait, the smallest β is 0.75 (Sect. 16.5.1), thus $V = 0.33U$ is the maximum speed. However, by using a duty factor of $\beta = 0.5$, we can achieve $V = U$, three times faster than the maximum speed of static walk. Figure 16.33 shows such a gait. The gait diagram (Fig. 16.33) shows that the robot stands on two legs at all times, which is called the *trot gait*.

As illustrated in Fig. 16.33b, a trot gait is a dynamic walk where the ZMP should be controlled to be inside the support pattern (support polygon) while the projection of the CoM goes outside of it. Hirose et al. introduced the body swaying control to meet the ZMP condition and the *expanded trot gait* for a smooth transition from a static gait at slow speed to a dynamic gait at high speed [16.60]. Later, this technique was elaborated for TITAN VI, the successor to TITAN IV [16.66].

16.5.3 Dynamic Quadrupeds Inspired by Mammals

Nature exhibits a variety of implementations of legged locomotion and some of them are much more dynamic than in the analysis and synthesis discussed above. Figure 16.34 shows two dynamic gaits used by mammals [16.67]. Pace is used by camels and bound is used by running dogs, for example.

Raibert showed that trot, pace, and bound can be regarded as biped locomotion by pairing diagonal legs, front–rear legs, and lateral legs, respectively. Using this idea, he demonstrated a hydraulic running quadruped that can trot, pace, and bound [16.5]. Such gaits have also been realized by electrically powered quadrupeds using different control schemes [16.68, 69].

Figure 16.35a shows recently developed mammal-like quadruped robots. Tekken is an electric quadruped robot with a 20 cm body length and weight of 3.1 kg developed by Fukuoka and Kimura [16.62]. It can walk on irregular terrain by using a controller based on a central pattern generator (CPG) and reflex controls.

BigDog is an energy-autonomous hydraulic quadruped robot that is 1 m tall and 1 m long, and weighs 90 kg, developed by Buehler et al., (Fig. 16.35b). Each leg has one passive linear pneumatic compliance in the lower leg, and three active joints for knee, hip pitch, and roll [16.63]. In an outdoor environment, BigDog can walk up and down 35° inclines, can trot at speeds of up to 0.8 m/s, and can carry over 50 kg of payload.

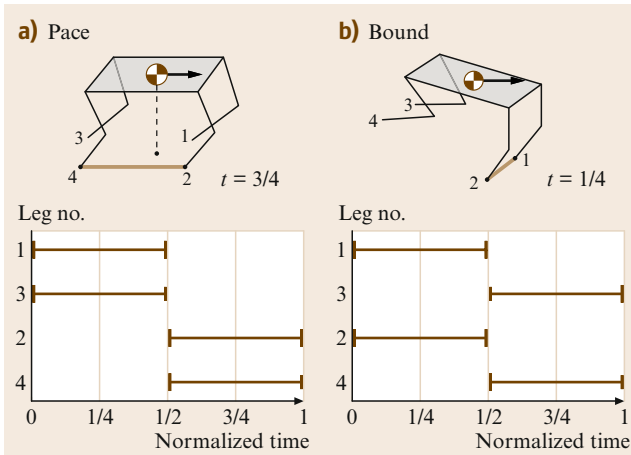


Fig. 16.34a,b Another dynamic gait of quadrupeds

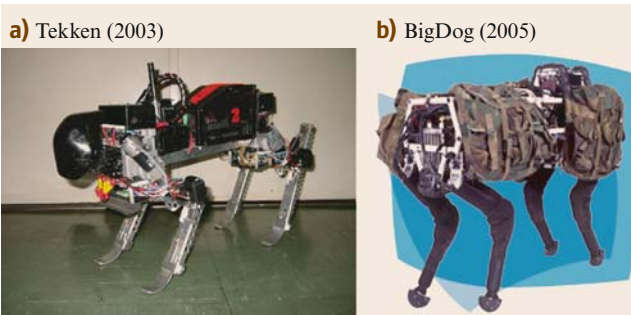


Fig. 16.35a,b Quadruped robots inspired by mammals [16.62, 63]

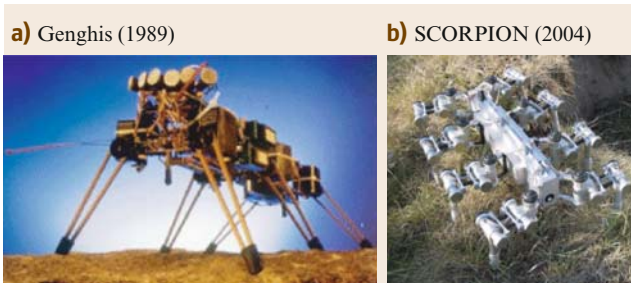


Fig. 16.36a,b Biologically inspired multilegged robots [16.64, 65]

16.5.4 Behavior-Based Multilegged Robots

Figure 16.36a shows a small hexapod robot Genghis (35 cm length and 1 kg weight) developed by Brooks [16.64]. Most importantly, the gait of Genghis was not explicitly controlled, but emerged from the carefully designed network named *subsumption architecture*. Genghis and other robots created by Brooks created a huge trend of *behavior-based*

robotics in robotics and artificial intelligence in the 1990s.

Figure 16.36b shows the eight-legged robot SCORPION (65 cm length and 11.5 kg weight) developed in collaboration between the University of Bremen and the National Aeronautics and Space Administration (NASA) [16.65]. The gait control of SCORPION is designed based on a behavior-based network and a central pattern generator.

16.6 Other Legged Robots

16.6.1 Leg-Wheel Hybrid Robots

Unlike animals or insects, a robot can be designed to have wheels that can rotate infinitely. By mixing the efficiency of wheels and the flexibility of legs, we can expect a robot of maximum terrain adaptivity with minimum power consumption. Figure 16.37 shows examples of such design concept.

Figure 16.37a shows stair climbing by a biped leg-wheeled robot developed by Matsumoto et al. [16.70]. The robot is a planer biped with telescopic legs, but the tip of each leg is equipped with a powered wheel. During the single-leg support phase, the robot is controlled as a wheeled inverted pendulum. In addition, a controller was developed to realize a smooth transient between the single support and statically stable double support phases.

Figure 16.37b shows RollerWalker, developed by Hirose et al. [16.71,76]. RollerWalker is a 12-DOF quadruped robot equipped with passive wheels on the tip of legs. It uses a roller-skating mode on a flat floor, while it can walk on an uneven terrain by retracting the passive wheels.

Figure 16.37c shows RHex developed by Buehler et al. [16.72]. Although it was originally inspired from the locomotion of cockroach, RHex has only six active DOF, that is, one actuator for each hip. Moreover, the legs can rotate full circle around the pitch axis. Using this unique design, RHex can walk and run over rugged, broken, and obstacle-ridden ground. Recently, it also demonstrated biped running with its rear legs [16.77].

Figure 16.37d shows Whigs II, another cockroach-inspired robot developed by Allen et al. [16.73]. This robot has only four active DOFs, one for propulsion, two for steering, and one for body flexion. Each leg is equipped with three spring-loaded spokes and is driven by the same actuator. Whigs II can realize

comparable mobility to RHex, while it uses fewer actuators.

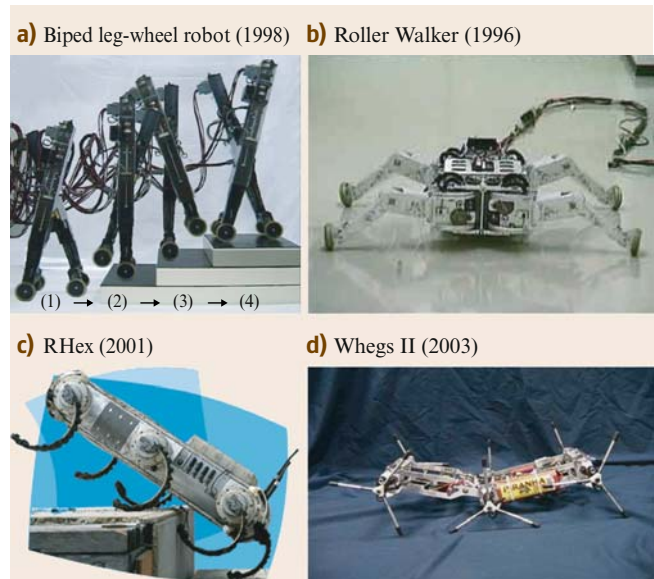


Fig. 16.37a–d Leg-wheel hybrid robots [16.70–73]

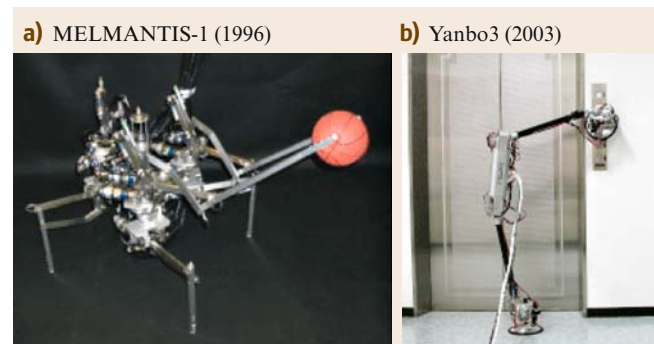
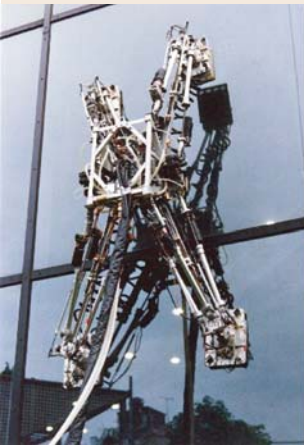


Fig. 16.38a,b Leg-arm hybrid robots [16.74,75]



Fig. 16.39 Dante II (1994) [16.78]

a) NINJA-1 (1991)



b) RiSE (2005)



Fig. 16.40a,b Wall-climbing robots [16.79, 80]

16.6.2 Leg-Arm Hybrid Robots

Another design concept is the leg-arm hybrid robot. Since legs have inherently many degree of freedom, it is possible to use them as manipulators. In this way, we can minimize the total number of DOFs, complexity,

weight, and power consumption of the walking robot. MELMANTIS-1 (Fig. 16.38a) developed by Koyachi et al. is a hexapod walker with 22 DOFs that can transform its legs into manipulators [16.74]. The robot can manipulate an object using two legs while standing with four other legs, after traveling on six legs for maximum stability.

Yanbo3 is a biped walker with eight DOFs developed by the group of Yoneda and Hirose [16.75]. It is designed to have the minimum number of DOFs necessary for a biped robot as well as for a manipulator when it is in the single support phase. In Fig. 16.38b, the robot is pressing the elevator button using its foot.

16.6.3 Tethered Walking Robots

Figure 16.39 shows Dante II, an eight-legged tethered walking robot developed by the eCMU Field Robotics Center in 1994. It was used at an Alaskan volcano for scientific exploration. To descend down steep crater walls in a rappelling-like manner, the robot uses a tether cable anchored at the crater rim [16.78]. Hirose et al. also developed a tethered quadruped for construction work [16.81].

16.6.4 Wall-Climbing Robots

Wall-climbing robots are characterized by their foot mechanisms and leg configurations. The vital part is the foot mechanism to generate the pulling force, and the use of vacuum suction cups, electromagnets (for steel wall), adhesive materials, or miniature spine array have been proposed.

Figure 16.40a shows the wall-climbing quadruped NINJA-1 developed by Hirose et al. [16.79]. Each foot of NINJA-1 is equipped with a specially designed suction pad that can minimize its vacuum leakage. Another reliable wall-climbing robot with suction cups was developed by Yano et al. [16.82].

Figure 16.40b shows the wall-climbing hexapod RiSE developed by Kim et al. [16.80, 83]. Each foot of RiSE is equipped with arrays of miniature spines observed in some insects and spiders. The robot can reliably climb on a wide variety of outdoor surfaces including concrete, stucco, brick, and dressed sandstone.

16.7 Performance Indices

In this section, we introduce useful performance indices that can be used to evaluate legged robots of different configurations.

16.7.1 Expansion of the Stability Margin Concept

As we see in Sect. 16.5.1, the stability margin was originally proposed for a degree of stability of the static walking multilegged robots by McGhee and Frank [16.57]. For a dynamic walking robot, we can define a stability margin as the minimum distance of the ZMP to the boundaries of the support polygon, since the ZMP is the natural extension of a projected CoM on the ground. This fact was already mentioned in the original work on the ZMP [16.36], and has been used implicitly by many researchers. Explicit definition of this ZMP stability margin can be seen, for example, in Huang et al. [16.41].

For a legged robot on rough terrain, Messuri and Klein defined the *energy stability margin* as the minimum potential energy required to tumble the robot as,

$$S_E = \min_i (Mgh_i), \quad (16.56)$$

where h_i is the CG height variation during the tumble around the i -th segment of the support polygon and M is the total mass of the robot [16.85]. This concept is widely accepted, and there are some proposals for improvement [16.31, 86].

16.7.2 Duty Factor and Froude Number

Throughout this chapter, we have observed various walking robots that might fit best for certain environments and purposes. In some cases, however, we need to compare walking robots which have different masses,

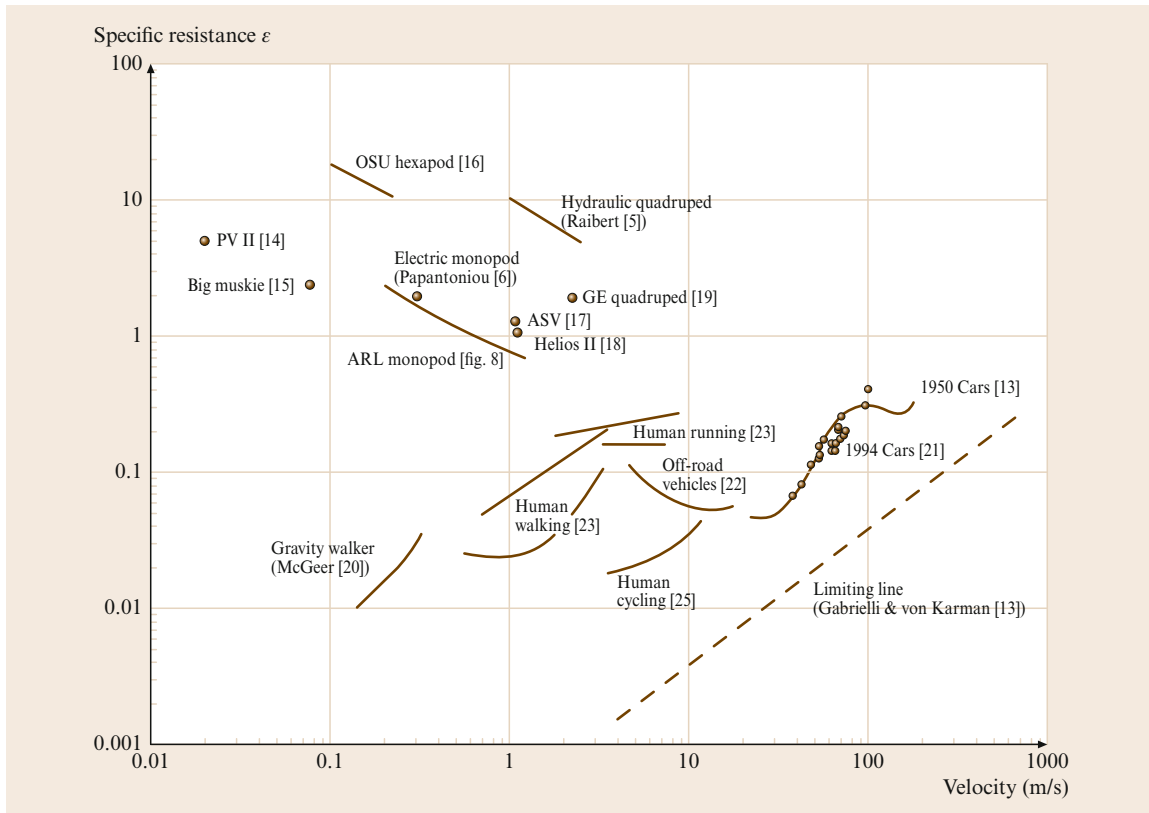


Fig. 16.41 The Gabrielli–von Karman diagram [16.84]

sizes, and numbers of legs by using a certain performance index. Such an index should be dimensionless like a Mach or Reynolds number in fluid mechanics.

One of the useful indices for walking machines was already shown, that is, the duty factor β defined as

$$\beta = \frac{(\text{support period})}{(\text{cycle time})}.$$

Duty factors can be used to make the distinction between walks and runs, since we have $\beta \geq 0.5$ for walking and $\beta < 0.5$ for running [16.67].

The Froude number is used in fluid mechanics to explain the behavior of surface waves. Since both of surface waves and legged locomotion are dynamic motion in gravity, *Alexander* used it to characterize animal locomotion [16.67, 87]. He calculated a Froude number by

$$F_{r2} = \frac{V^2}{gh}, \quad (16.57)$$

where V is the walking or running speed, g is the acceleration due to gravity, and h is the height of hip joint from the ground. He showed that animals of different sizes use similar gaits when they travel with equal Froude numbers. In particular, most animals change their gait from walking to running at a speed equivalent to a Froude number of $F_{r2} = 1$.

The Froude number is also defined as

$$F_{r1} = \frac{V}{\sqrt{gh}}, \quad (16.58)$$

16.8 Conclusions and Future Trends

In this chapter, we have discussed the following topics.

- Sect. 16.1 *A brief history:* The history of legged robot research was introduced.
- Sect. 16.2 *Analysis of cycling walking:* As the typical cycling walking robot, a simple passive walker was analyzed. The Poincaré map is one of the most important tools for this purpose.
- Sect. 16.3 *Control of biped robots using forward dynamics:* The dynamics and control of biped robots were discussed. The treatment of the unilateral ground force is the key issue for bipeds, as well as other legged robots.

which is the square root of F_{r2} and can be used as a dimensionless speed for animals or legged robots.

16.7.3 Specific Resistance

The *specific resistance* is another important dimensionless number that is used to evaluate the energy efficiency of a mobile robot.

Gabrielli and von Karman discussed the performance of various vehicles using the power consumption per unit distance. That is

$$\epsilon = \frac{E}{Mgd}, \quad (16.59)$$

where E is the total energy consumption for a travel of distance d , M is the total mass of the vehicle, and g is the acceleration due to gravity [16.88]. Note that when we push a box of mass M a distance d on a floor with friction coefficient μ , we consume the energy $Mg\mu d$ and the specific resistance becomes $\epsilon = \mu$. Therefore, we can say that the specific resistance indicates how smooth the locomotion is.

In their original work, Gabrielli and von Karman plotted the specific resistance as a function of speed for various vehicles as Fig. 16.41. This is called the *Gabrielli–von Karman diagram* and it was used to compare various styles of locomotion by *Umetani* and *Hirose* [16.89]. *Gregorio*, *Ahmadi* and *Buehler* also showed the specific resistance of recent walking robots including their efficient hopping robot, ARL monomod [16.84].

- Sect. 16.4 *Biped robots in the ZMP scheme:* As a practical scheme to control biped robots, the zero-moment point (ZMP) was discussed.
- Sect. 16.5 *Multilegged robots:* The relationship between gaits and stability was discussed. In addition, the landmark robots in this field were introduced.
- Sect. 16.6 *Other legged robots:* We introduced leg–wheel hybrid, leg–arm hybrid, tethered walking, and wall-climbing robots. One might be impressed with the range of imagination displayed here, but these examples are still just the tip of the iceberg.

Sect. 16.7 *Performance indices*: The stability margin, Froude number, and the specific resistance were discussed. They are useful performance indices to compare legged robots in different configurations.

Over the last three decades, legged robots have become faster, more efficient, and more robust. This trend may continue in the future, aiming for ultimate mobility comparable to that of insects, mammals, and ourselves.

References

- 16.1 M. Wisse, L. Schwab, F.L.T. Van der Helm: Passive walking dynamic model with upper body, *Robotica* **22**(6), 681–688 (2004)
- 16.2 C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E.R. Westervelt, C. Canudas-de-Wit, J.W. Grizzle: RABBIT: a testbed for advanced control theory, *IEEE Contr. Syst. Mag.* **23**(5), 57–79 (2003)
- 16.3 M. Vukobratović, B. Borovac: Zero-moment point – Thirty five years of its life, *Int. J. Humanoid Robot.* **1**(1), 157–173 (2004)
- 16.4 T. McGeer: Passive dynamic walking, *Int. J. Robot. Res.* **9**(2), 62–82 (1990)
- 16.5 M.H. Raibert: *Legged Robots That Balance* (MIT Press, Cambridge 1986)
- 16.6 M. Coleman, A. Ruina: An uncontrolled walking toy that cannot stand still, *Phys. Rev. Lett.* **80**(16), 3658–3661 (1998)
- 16.7 A. Goswami, B. Thuilot, B. Espiau: A study of a compass-like biped robot: symmetry and chaos, *Int. J. Robot. Res.* **17**(12), 1282–1301 (1998)
- 16.8 C. Azevedo, B. Amblard, B. Espiau, C. Assaïante: A synthesis of bipedal locomotion in human and robots, *Res. Rep.* 5450, INRIA <https://hal.inria.fr/inria-00070557> (December 2004)
- 16.9 C. Azevedo, B. Espiau, B. Amblard, C. Assaïante: Bipedal locomotion: toward unified concepts in robotics and neurosciences, *Biol. Cybern.* **96**(2), 209–228 (2007)
- 16.10 P.B. Wieber: Constrained stability and parameterized control in biped walking, *Int. Symp. Math. Theory Netw. Syst.* (2000)
- 16.11 P.B. Wieber: On the stability of walking systems, *Int. Workshop Humanoids Human Friendly Robot.* (2002)
- 16.12 F. Pfeiffer, C. Glocker: *Multibody Dynamics with Unilateral Contacts* (Wiley, New York 1996)
- 16.13 L. Righetti, A.J. Ijspeert: Programmable central pattern generators: an application to biped locomotion control, *IEEE Int. Conf. Robot. Autom.* (Orlando, USA May 2006)
- 16.14 K. Matsuoka: Sustained oscillations generated by mutually inhibiting neurons with adaptation, *Biol. Cybern.* **52**, 345–353 (1985)
- 16.15 G. Endo, J. Morimoto, J. Nakanishi, G. Cheng: An empirical exploration of a neural oscillator for biped locomotion control, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp. 3036–3042
- 16.16 D.C. Witt: A feasibility study on powered lower-limb prosthesis, *Univer. Oxford Dep. Eng. Sci. Rep.* (1970)
- 16.17 F. Gubina, H. Hemami, R.B. McGhee: On the dynamic stability of biped locomotion, *IEEE Trans. Biomed. Eng.* **BME-21**(2), 102–108 (1974)
- 16.18 H. Miura, I. Shimoyama: Dynamic walk of a biped, *The Int. J. Robot. Res.* **3**(2), 60–74 (1984)
- 16.19 J. Furusho, M. Masubuchi: A theoretically motivated reduced order model for the control of dynamic biped locomotion, *J. Dyn. Syst. Meas. Contr.* **109**, 155–163 (1987)
- 16.20 S. Kawamura, F. Miyazaki, S. Arimoto: Realization of robot motion based on a learning method, *IEEE Trans. Syst. Man Cybern.* **18**(1), 126–134 (1988)
- 16.21 S. Kajita, T. Yamaura, A. Kobayashi: Dynamic walking control of a biped robot along a potential energy conserving orbit, *IEEE Trans. Robot. Autom.* **8**(4), 431–438 (1992)
- 16.22 J. Pratt, C.-M. Chew, A. Torres, P. Dilworth, G. Pratt: An intuitive approach for bipedal locomotion, *Int. J. Robot. Res.* **20**(2), 129–143 (2001)
- 16.23 E.R. Westervelt, J.W. Grizzle, D.E. Koditschek: Hybrid zero dynamics of planar biped walkers, *IEEE Trans. Autom. Contr.* **48**(1), 42–56 (2003)
- 16.24 S. Lohmeier, K. Löffler, M. Gienger, H. Ulbrich, F. Pfeiffer: Computer system and control of biped Johnnie, *IEEE Int. Conf. Robot. Autom.* (New-Orleans 2004) pp. 4222–4227
- 16.25 J.H. Kim, J.H. Oh: Walking control of the humanoid platform KHR-1 based on torque feedback control, *IEEE Int. Conf. Robot. Autom.* (2004) pp. 623–628
- 16.26 P.B. Wieber, C. Chevallereau: Online adaptation of reference trajectories for the control of walking systems, *Robot. Auton. Syst.* **54**(7), 559–566 (2006)
- 16.27 F. Allgöwer, T.A. Badgwell, J.B. Rawlings, S.J. Wright: Nonlinear predictive control and moving horizon estimation an overview., *Eur. Contr. Conf.* (Karlsruhe 1999) pp. 392–449
- 16.28 C. Azevedo, P. Poignet, B. Espiau: Artificial locomotion control: from human to robots, *Robot. Auton. Syst.* **47**(4), 203–223 (2004)
- 16.29 P.B. Wieber: Trajectory-free linear model predictive control for stable walking in the presence

- of strong perturbations, IEEE-RAS Int. Conf. Humanoid Robots (Genoa 2006)
- 16.30 A. Takanishi, M. Ishida, Y. Yamazaki, I. Kato: The realization of dynamic walking by the biped walking robot WL-10RD, Int. Conf. Adv. Robot. (ICAR'85) (1985) pp. 459–466
- 16.31 M. Hirose, Y. Haikawa, T. Takenaka, K. Hirai: Development of humanoid robot ASIMO, IEEE/RSJ Int. Conf. Intell. Robots Syst. – Workshop 2 (2001)
- 16.32 M. Gienger, K. Löffler, F. Pfeiffer: Towards the design of a biped jogging robot, IEEE Int. Conf. Robot. Autom. (2001) pp. 4140–4145
- 16.33 K. Kaneko, S. Kajita, F. Kanehiro, K. Yokoi, K. Fujiwara, H. Hirukawa, T. Kawasaki, M. Hirata, T. Isozumi: Design of advanced leg module for humanoid robotics project of METI, IEEE Int. Conf. Robot. Autom. (2002) pp. 38–45
- 16.34 Y. Sugahara, M. Kawase, Y. Mikuriya, T. Hosobata, H. Sunazuka, K. Hashimoto, H. Lim, A. Takanishi: Support torque reduction mechanism for biped locomotor with parallel mechanism, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2004) pp. 3213–3218
- 16.35 I.W. Park, J.Y. Kim, J. Lee, J.H. Oh: Online free walking trajectory generation for biped humanoid robot KHR-3(HUBO), IEEE Int. Conf. Robot. Autom. (Orlando 2006) pp. 1231–1236
- 16.36 M. Vukobratović, J. Stepanenko: On the stability of anthropomorphic systems, Math. Biosci. **15**, 1–37 (1972)
- 16.37 S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Biped walking pattern generation by using preview control of zero-moment point, IEEE Int. Conf. Robot. Autom. (2003) pp. 1620–1626
- 16.38 A. Takanishi, Y. Egusa, M. Tochizawa, T. Takeya, I. Kato: Realization of dynamic biped walking stabilized with trunk motion, ROMANSY 7 (1988) pp. 68–79
- 16.39 A. Takanishi, H. Lim, M. Tsuda, I. Kato: Realization of dynamic biped walking stabilized by trunk motion on a sagittally uneven surface, IEEE Int. Workshop Intell. Robots Syst. (1990) pp. 323–330
- 16.40 S. Kagami, K. Nishiwaki, T. Kitagawa, T. Sugihara, M. Inaba, H. Inoue: A fast generation method of a dynamically stable humanoid robot trajectory with enhanced ZMP constraint, IEEE Int. Conf. Humanoid Robot. (2000)
- 16.41 Q. Huang, K. Yokoi, S. Kajita, K. Kaneko, H. Arai, N. Koyachi, K. Tanie: Planning walking patterns for a biped robot, IEEE Trans. Robot. Autom. **17**(3), 280–289 (2001)
- 16.42 T. Sugihara, Y. Nakamura, H. Inoue: Realtime humanoid motion generation through ZMP manipulation based on inverted pendulum control, IEEE Int. Conf. Robot. Autom. (2002) pp. 1404–1409
- 16.43 K. Nagasaka, K. Kuroki, S. Suzuki, Y. Itoh, J. Yamaguchi: Integrated motion control for walking, jumping and running on a small bipedal entertainment robot, IEEE Int. Conf. Robot. Autom. (2004) pp. 3189–3194
- 16.44 K. Harada, S. Kajita, F. Kanehiro, K. Fujiwara, K. Kaneko, K. Yokoi, H. Hirukawa: Real-time planning of humanoid robot's gait for force controlled manipulation, IEEE Int. Conf. Robot. Autom. (2004) pp. 616–622
- 16.45 M. Tomizuka, D.E. Rosenthal: On the optimal digital state vector feedback controller with integral and preview actions, Trans. the ASME J. Dyn. Syst. Meas. Contr. **101**, 172–178 (1979)
- 16.46 T. Katayama, T. Ohki, T. Inoue, T. Kato: Design of an optimal controller for a discrete time system subject to previewable demand, Int. J. Contr. **41**(3), 677–699 (1985)
- 16.47 J. Yamaguchi, A. Takanishi, I. Kato: Experimental development of a foot mechanism with shock absorbing material for acquisition of landing surface position information and stabilization of dynamic biped walking, IEEE Int. Conf. Robot. Autom. (1995) pp. 2892–2899
- 16.48 K. Hirai, M. Hirose, Y. Haikawa, T. Takenaka: The development of honda humanoid robot, IEEE Int. Conf. Robot. Autom. (1998) pp. 1321–1326
- 16.49 K. Yokoi, F. Kanehiro, K. Kaneko, S. Kajita, K. Fujiwara, H. Hirukawa: Experimental study of humanoid robot HRP-1S, Int. J. Robot. Res. **23**(4–5), 351–362 (2004)
- 16.50 K. Hashimoto, Y. Sugahara, H. Sunazuka, C. Tanaka, A. Ohta, M. Kawase, H. Lim, A. Takanishi: Biped landing pattern modification method with nonlinear compliance control, IEEE Int. Conf. Robot. Autom. (Orlando 2006) pp. 1213–1218
- 16.51 A. Goswami: Postural stability of biped robots and the foot-rotation indicator(FRI) point, Int. J. Robot. Res. **18**(6), 523–533 (1999)
- 16.52 P. Sardain, G. Bessonnet: Forces acting on a biped robot. center of pressure-zero moment point, IEEE Trans. Syst. Man Cybern. Part A: Syst. Humans **34**(5), 630–637 (2004)
- 16.53 T. Saida, Y. Yokokoji, T. Yoshikawa: FSW (feasible solution of wrench) for multi-legged robots, IEEE Int. Conf. Robot. Autom. (2003) pp. 3815–3820
- 16.54 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots – Adios ZMP, IEEE Int. Conf. Robot. Autom. (Orlando 2006), pp. 1976–1983
- 16.55 R.B. McGhee: Vehicular legged locomotion. In: *Advances in Automation and Robotics*, ed. by G.N. Saridis (JAI Press, New York 1985) pp. 259–284
- 16.56 S.M. Song, K.J. Waldron: *Machines that Walk: the Adaptive Suspension Vehicle* (The MIT Press, Cambridge 1989)
- 16.57 R.B. McGhee, A.A. Frank: On the stability properties of quadruped creeping gaits, Math. Biosci. **3**, 331–351 (1968)

- 16.58 K.J. Waldron, R.B. McGhee: The adaptive suspension vehicle, *IEEE Contr. Syst. Mag.* **6**, 7–12 (1986)
- 16.59 S. Hirose, Y. Fukuda, H. Kikuchi: The gait control system of a quadruped walking vehicle, *Adv. Robot.* **1**(4), 289–323 (1986)
- 16.60 S. Hirose, K. Yoneda, R. Furuya, T. Takagi: Dynamic and static fusion control of quadruped walking vehicle, *IEEE/RSJ Int. Workshop Intell. Robots Syst.* (1989) pp. 199–204
- 16.61 S. Hirose: A study of design and control of a quadruped walking vehicle, *Int. J. Robot. Res.* **3**(2), 113–133 (1984)
- 16.62 Y. Fukuoka, H. Kimura, A.H. Cohen: Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts, *Int. J. Robot. Res.* **22**(3–4), 187–202 (2003)
- 16.63 M. Buehler, R. Playter, M. Raibert: Robots step outside, *Int. Symp. Adapt. Motion Animals Mach. (AMAM)* (Ilmenau 2005)
- 16.64 R.A. Brooks: A robot that walks; emergent behavior from a carefully evolved network, *IEEE Int. Conf. Robot. Autom.* (Scottsdale 1989) pp. 292–296
- 16.65 D. Spennberg, K. McCullough, F. Kirchner: Stability of walking in a multilegged robot suffering leg loss, *IEEE Int. Conf. Robot. Autom.* (2004) pp. 2159–2164
- 16.66 S. Hirose, K. Yoneda, K. Arai, T. Ibe: Design of prismatic quadruped walking vehicle TITAN VI, 5th Int. Conf. Adv. Robot. (Pisa Italy 1991) pp. 723–728
- 16.67 R. McNeill Alexander: The gait of bipedal and quadrupedal animals, *Int. J. Robot. Res.* **3**(2), 49–59 (1984)
- 16.68 H. Kimura, I. Shimoyama, H. Miura: Dynamics in the dynamic walk of a quadruped robot, *Adv. Robot.* **4**(3), 283–301 (1990)
- 16.69 J. Furusho, A. Sano, M. Sakaguchi, E. Koizumi: Realization of bounce gait in a quadruped robot with articular-joint-type legs, *IEEE Int. Conf. Robot. Autom.* (1995) pp. 697–702
- 16.70 O. Matsumoto, S. Kajita, M. Saigo, K. Tani: Dynamic trajectory control of passing over stairs by a biped type leg-wheeled robot with nominal reference of static gait, *IEEE/RSJ Int. Conf. Intell. Robot Syst.* (1998) pp. 406–412
- 16.71 S. Hirose, H. Takeuchi: Study on roller-walk (basic characteristics and its control), *IEEE Int. Conf. Robot. Autom.* (1996) pp. 3265–3270
- 16.72 U. Saranli, M. Buehler, D.E. Koditschek: RHex: a simple and highly mobile hexapod robot, *Int. J. Robot. Res.* **20**(7), 616–631 (2001)
- 16.73 T.J. Allen, R.D. Quinn, R.J. Bachmann, R.E. Ritzmann: Abstracted biological principles applied with reduced actuation improve mobility of legged vehicles, *IEEE Int. Conf. Intell. Robots Syst.* (Las Vegas 2003) pp. 1370–1375
- 16.74 N. Koyachi, H. Adachi, M. Izumi, T. Hirose, N. Senjo, R. Murata, T. Arai: Multimodal control of hexapod mobile manipulator MELMANTIS-1, 5th Int. Conf. Climbing Walking Robots (2002) pp. 471–478
- 16.75 Y. Ota, T. Tamaki, K. Yoneda, S. Hirose: Development of walking manipulator with versatile locomotion, *IEEE Int. Conf. Robot. Autom.* (2003) pp. 477–483
- 16.76 G. Endo, S. Hirose: Study on roller-walker: system integration and basic experiments, *IEEE Int. Conf. Robot. Autom.* (Detroit 1999) pp. 2032–2037
- 16.77 N. Neville, M. Buehler, I. Sharf: A bipedal running robot with one actuator per leg, *IEEE Int. Conf. Robot. Autom.* (Orlando 2006) pp. 848–853
- 16.78 J. Bares, D. Wettergreen: Dante II: technical description, results and lessons learned, *Int. J. Robot. Res.* **18**(7), 621–649 (1999)
- 16.79 S. Hirose, A. Nagakubo, R. Toyama: Machine that can walk and climb on floors, walls and ceilings, 5th Int. Conf. Adv. Robot. (Pisa 1991) pp. 753–758
- 16.80 S. Kim, A. Asbeck, W. Provancher, M.R. Cutkosky: SpinybotII: Climbing hard walls with compliant microspines, *IEEE ICAR* (Seattle 2005) pp. 18–20
- 16.81 S. Hirose, K. Yoneda, H. Tsukagoshi: TITAN VII: quadruped walking and manipulating robot on a steep slope, *IEEE Int. Conf. Robot. Autom.* (1997)
- 16.82 T. Yano, S. Numao, Y. Kitamura: Development of a self-contained wall climbing robot with scanning type suction cups, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (1998) pp. 249–254
- 16.83 A.T. Asbeck, S. Kim, A. McClung, A. Parness, M.R. Cutkosky: Climbing walls with microspines (video), *IEEE Int. Conf. Robot. Autom.* (Orlando 2006)
- 16.84 P. Gregorio, M. Ahmadi, M. Buehler: Design, control, and energetics of an electrically actuated legged robot, *IEEE Trans. Syst. Man Cyber. – Part B: Cyber.* **27**(4), 626–634 (1997)
- 16.85 D.A. Messuri, C.A. Klein: Automatic body regulation for maintaining stability of a legged vehicle during rough-terrain locomotion, *IEEE J. Robot. Autom.* **RA-1**(3), 132–141 (1985)
- 16.86 E. Garcia, P. Gonzalez de Santos: An improved energy stability margin for walking machines subject to dynamic effects, *Robotica* **23**(1), 13–20 (2005)
- 16.87 R. McNeill Alexander: *Exploring Biomechanics – Animals in Motion* (W.H. Freeman, New York 1992)
- 16.88 G. Gabrielli, T. von Karman: What price speed – specific power required for propulsion of vehicles, *Mech. Eng.* **72**(10), 775–781 (1950)
- 16.89 Y. Umetani, S. Hirose: Biomechanical study of serpentine locomotion (evaluation as a locomotion measure) (in Japanese). In: *BIOMECHANISM*(2) (Univ. Tokyo Press, Tokyo 1973) pp. 289–297

Wheeled Robots

Guy Campion, Woojin Chung

The purpose of this chapter is to introduce, analyze, and compare the models of wheeled mobile robots (WMR) and to present several realizations and commonly encountered designs. The mobility of WMR is discussed on the basis of the kinematic constraints resulting from the pure rolling conditions at the contact points between the wheels and the ground. According to this discussion it is shown that, whatever the number and the types of the wheels, all WMR belong to only five generic classes. Different types of models are derived and compared: the posture model versus the configuration model, the kinematic model versus the dynamic model. The structural properties of these models are discussed and compared. These models as well as their properties constitute the background necessary for model-based control design. Practical robot structures are classified according to the number of wheels, and features are introduced focusing on commonly adopted designs. Omnimobile robots and articulated robots realizations are described in more detail.

17.1 Overview	391
17.2 Mobility of Wheeled Robots	392
17.2.1 Types of Wheels	392
17.2.2 Kinematic Constraints	394
17.2.3 Robot Configuration Variables.....	396
17.2.4 Restriction on Robot Mobility.....	396

17.2.5 Characterization of Robot Mobility..	397
17.2.6 The Five Classes of Wheeled Mobile Robots	398
17.3 State-Space Models of Wheeled Mobile Robots	398
17.3.1 Posture Kinematic Models	398
17.3.2 Configuration Kinematic Models.....	399
17.3.3 Configuration Dynamic Models.....	400
17.3.4 Posture Dynamic Models	401
17.3.5 Articulated Robots.....	401
17.4 Structural Properties of Wheeled Robots Models	403
17.4.1 Irreducibility, Controllability, and Nonholonomy	403
17.4.2 Stabilizability	404
17.4.3 Static State-Feedback Linearizability.....	404
17.4.4 Dynamic State-Feedback Linearizability – Differential Flatness	404
17.5 Wheeled Robot Structures	405
17.5.1 Robots with One Wheel	405
17.5.2 Robots with Two Wheels	405
17.5.3 Robots with Three Wheels	406
17.5.4 Four Robots with Four Wheels.....	408
17.5.5 Special Applications of Wheeled Robots	408
17.6 Conclusions	409
References	410

17.1 Overview

The purpose of this chapter is to provide a general description of wheeled mobile robots, to discuss their properties from the mobility point of view, to introduce several dynamical models necessary for the design of model-based control laws, and to describe the most commonly encountered realizations of such robots.

Throughout the chapter we make the assumption that the wheels satisfy the kinematic constraints relative to the pure rolling conditions at each contact wheel/ground, without sliding effects. This implies that we assume that the contact forces between the ground and the wheels magically take the right values allow-

ing the satisfaction of these conditions; this is an ideal model. In reality the contact forces appear as a consequence of local sliding, according to phenomenological contact force models. Using a singular perturbation approach it can be shown however that these sliding effects correspond to fast dynamics, i.e., to dynamical effects with characteristic times that are quite short with respect to the dynamics of the global motion of the robot, and can therefore be neglected, at least when using the ideal model for control design purpose [17.1] (Chap. 34).

The chapter is organized as follows. Section 17.2 is devoted to the characterization of the restriction of robot motion induced by these pure rolling conditions. We first describe the different types of wheels used in the construction of mobile robots and derive the corresponding kinematic constraints. This allows us to characterize the mobility of a robot equipped with several wheels of these different types, and we show that these robots can be classified into only five categories, corresponding to two mobility indices.

17.2 Mobility of Wheeled Robots

In this section we describe a variety of wheels and wheel implementations in mobile robots. We discuss the restriction of robot mobility implied by the use of these wheels and deduce a classification of robot mobility allowing one to characterize robot mobility fully, whatever the number and type of the wheels.

17.2.1 Types of Wheels

In order to achieve robot locomotion, wheeled mobile robots are widely used in many applications. In general, wheeled robots consume less energy and move faster than other locomotion mechanisms (e.g., legged robots or tracked vehicles). From the viewpoint of control, less control effort is required, owing to their simple mechanisms and reduced stability problems. Although it is difficult to overcome rough terrain or uneven ground conditions, wheeled mobile robots are suitable for a large class of target environments in practical applications. When we think of a single-wheel design, there are two candidates: a standard wheel or a special wheel. A standard wheel can be understood as a conventional tire. Special wheels possess unique mechanical structures including rollers or spheres. Figure 17.1 shows the general

In Sect. 17.3, we present four types of generic state-space models allowing one to describe robot behavior within each of these five categories, and the relationships between these models. We introduce *kinematic* and *dynamic* models, whose inputs are, respectively, velocities and accelerations (or, equivalently input torques), as well as *posture* or *configuration* models, corresponding to a minimal description of the robot behavior, or to a full description, including the internal variables, respectively.

In Sect. 17.4, we present several structural properties of these models, from a control design point of view. We first discuss the questions of stabilizability, controllability, and nonholonomy of restricted mobility robots. We then discuss the problem of state feedback linearization, either input–output linearization by static state feedback, or full linearization by dynamic extension and dynamic state feedback.

In the last section we present several realizations of wheeled mobile robots, with several particular devices such as synchronous drive, Swedish wheels, and articulated robots.

design of a standard wheel. Three conditions should be defined for a standard wheel design:

1. the determination of the two offsets d and b
2. a mechanical design that allows steering motion or not (i.e., to fix the wheel orientation or not)
3. the determination of steering and driving actuation (i.e., active or passive drive)

Condition 1 is the kinematic parameter design problem for a single standard wheel. The parameter d can be either 0 or some positive constant. Parameter b is the lateral offset of the wheel and is usually set to zero. In a special design, a nonzero b may be selected to obtain pure rolling contact between the wheel and ground without causing rotational slip at the contact point. However, this is rarely used and we mainly consider the case of zero lateral offset b .

Condition 2 is a design problem for whether the wheel orientation can be changed or not. If the steering axis is fixed, the wheel provides a velocity constraint on the driving direction. Condition 3 is the design problem of whether to actuate steering or driving motion by actuators or to drive steering or motion passively.

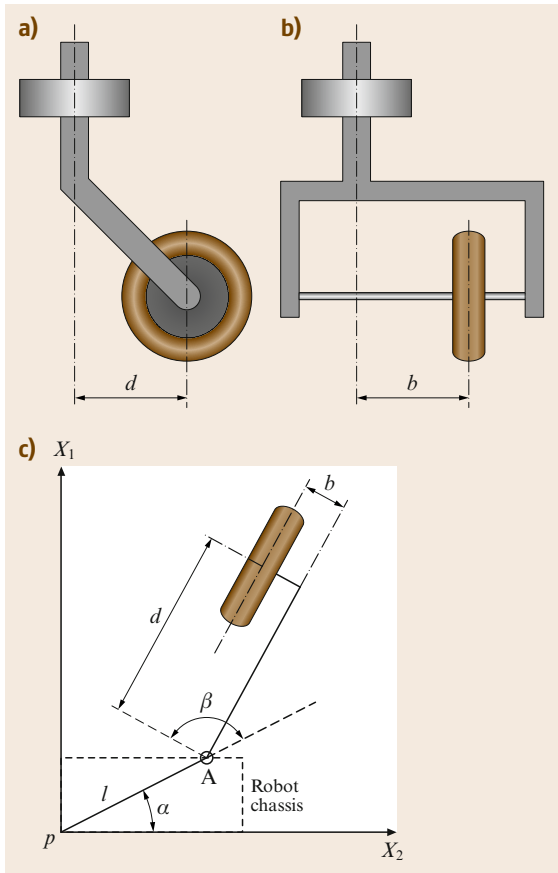


Fig. 17.1a-c The general design of a standard wheel. (a) side view (b) front view (c) top view

If steering motion is allowed, the offset d plays a significant role in the kinematic modeling. For a conventional caster wheel (i.e., an off-centered orientable wheel), there is a nonzero offset d . Point A in Fig. 17.1 indicates the location of the joint connecting the wheel module to the robot chassis. Two orthogonal linear velocity components at point A are obtained by a caster wheel, which result from the steering and driving motions of the wheel module. This implies that a passive caster wheel does not provide an additional velocity constraint on the robot's motion. If a caster wheel is equipped with two actuators that drive steering and driving motions independently, holonomic omnidirectional movement can be achieved because any desired velocity at point A can be generated by solving the inverse kinematics problem.

If the offset d is set to zero, the allowable velocity direction at point A is limited to the wheel orientation.

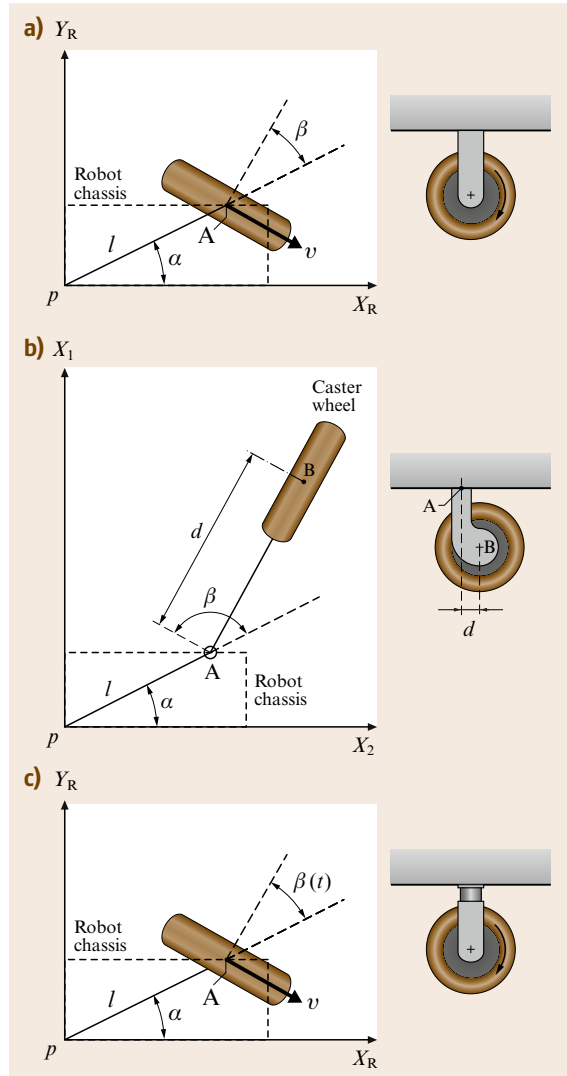
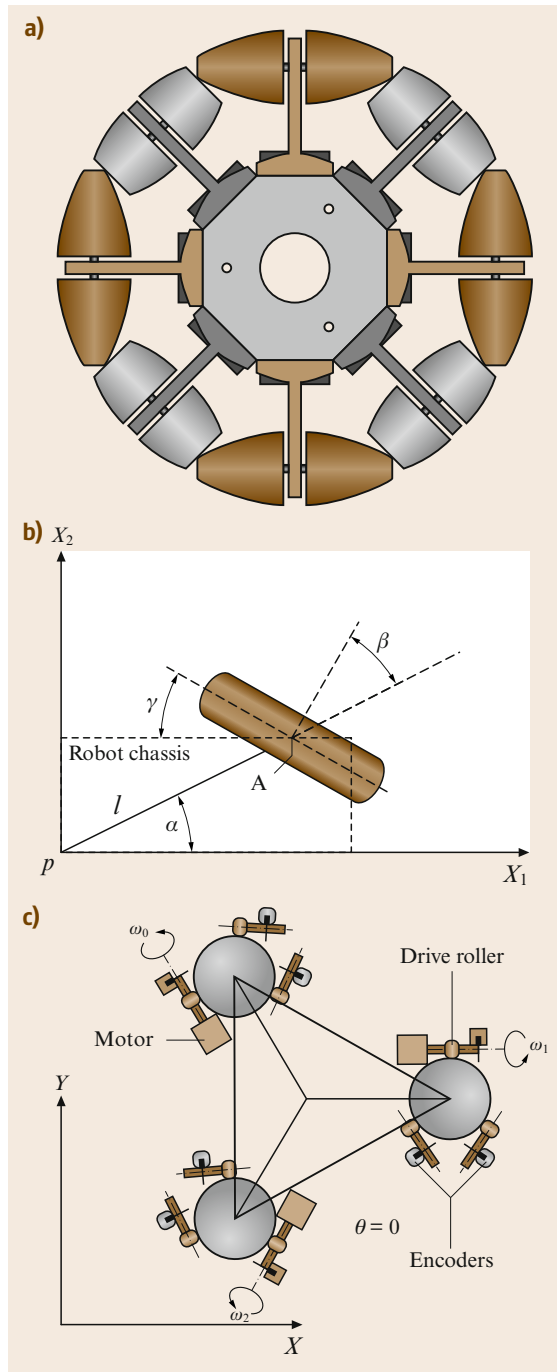


Fig. 17.2a-c Structures of standard wheels. (a) Passive fixed wheel. (b) Passive or active, off-centered orientable wheel. (c) Active orientable wheel without offsets

In such a case, the steering motion should not be passive because the wheel orientation cannot be changed passively. However, the driving velocity can be determined passively by actuation of other wheels. Wheel orientation should be actively steered to the desired velocity direction due to the nonholonomic velocity constraint. This implies that the wheel orientation should be aligned before movement.

In summary, four types of standard wheels are commonly used. The first is a passively driven wheel with



a fixed steering axis. The second is a passive caster wheel with offset d . The third is an active caster wheel with offset d , where the steering and driving motions are con-

Fig. 17.3 (a) Swedish wheel. (b) Attachment of a Swedish wheel and (c) spherical wheel [17.2] ◀

trolled by actuators. The fourth is an active orientable wheel with zero offset d , where steering and driving motions are driven by actuators. The structures of each wheel type are shown in Fig. 17.2. The kinematics and constraints of those wheels will be explained in detail in Sect. 17.2.2.

Although standard wheels are advantageous because of their simple structure and good reliability, the nonholonomic velocity constraint (i. e., no side-slip condition) limits robot motion. On the other hand, special wheels can be employed in order to obtain omnidirectional motion of a mobile robot (omnimobile robot), i. e., to ensure three degrees of freedom for plane motion. We consider two typical designs of special wheels: the Swedish wheel and the spherical wheel. Figure 17.3a shows the Swedish wheel. Small passive free rollers are located along the outer rim of the wheel. Free rollers are employed in order to eliminate the nonholonomic velocity constraint. Passive rollers are free to rotate around the axis of rotation, which results in lateral motion of the wheel. As a result, a driving velocity should be controlled, while the lateral velocity is passively determined by the actuation of the other wheels.

A spherical wheel is shown in Fig. 17.3c. The rotation of the sphere is constrained by rollers that make rolling contact with the sphere. The rollers can be divided into driving and supporting rollers. The sphere is driven by actuation of the driving rollers, whereas the rolling contacts provide nonholonomic constraints, and the resultant motion of the sphere module becomes holonomic. This implies that the robot can be moved with any desired linear/angular velocities at any time. By using the spherical wheel, a holonomic omnidirectional mobile robot can be developed and the robot achieves smooth and continuous contact between the sphere and the ground. However, the design of the sphere-supporting mechanism is difficult and the payload must be quite low due to the point contact. Another drawback is that the surface of the sphere can be polluted when traveling over dirty ground and it is difficult to overcome irregular ground conditions. These drawbacks limit the practical application of the spherical wheel. An example of the use of spherical wheels can be found in [17.2] and [17.3]. The spherical structure can also be applied to special robotic transmissions; examples include the nonholonomic manipulator in [17.4] and the passive haptic system in [17.5].

17.2.2 Kinematic Constraints

We assume, as a first step, that the mobile robot under study is made up of a rigid cart equipped with non-deformable wheels, and that it is moving on a horizontal plane. The position of the robot on the plane is described, with respect to an arbitrary inertial frame, by the *posture* vector $\xi = (x \ y \ \theta)^T$, where x and y are the coordinates of a reference point P of the robot cart, while θ describes the orientation of a mobile frame attached to the robot, with respect to the inertial frame (Fig. 17.4).

We assume that, during motion, the plane of each wheel remains vertical and the wheel rotates around its horizontal axle, whose orientation with respect to the cart can be fixed or varying. We distinguish between two basic classes of idealized wheels, namely conventional and the Swedish wheels. In each case, it is assumed that the contact between the wheel and the ground is reduced to a single point. The kinematic constraints result from the fact that the velocity of the material point of the wheel in contact with the ground is equal to zero.

For a conventional wheel, the kinematic constraints imply that the velocity of the center of the wheel is parallel to the wheel plane (*nonslip condition*) and is proportional to the wheel rotation velocity (*pure rolling condition*). For each wheel the kinematic constraints therefore result in two independent conditions. For a Swedish wheel, due to the relative rotation of the rollers with respect to the wheel, only one of the velocity components of the wheel contact point is zero. The direction of this zero component is fixed with respect to the wheel plane and depends on the wheel construction.

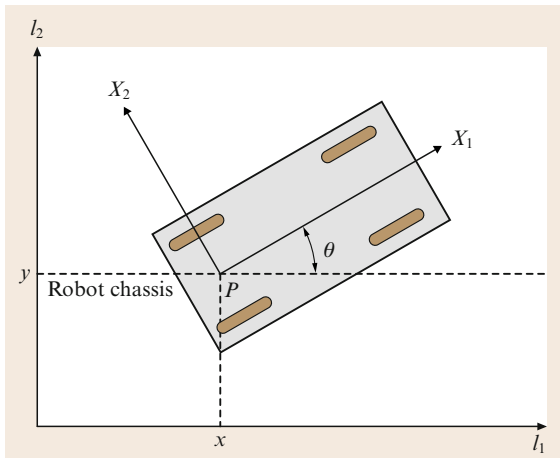


Fig. 17.4 The posture definition of a mobile robot on a plane

For such wheels the kinematic constraints result in only one condition.

Conventional Wheels

We now derive the general form of the kinematic constraints for a conventional wheel.

As shown in Fig. 17.2, there are several variations of the conventional wheel design. First, we focus on the off-centered orientable wheel in Fig. 17.2b. The center of the wheel, B, is connected to the cart by a rigid rod from A (a fixed point on the cart) to B, aligned with the wheel plane. The rod, whose length is denoted by d , can rotate around a fixed vertical axle at point A. The position of A is specified by two constant polar coordinates, l and α , with respect to the reference point P. The rotation of the rod with respect to the cart is represented by the angle β . The radius of the wheel is denoted by r , and its angle of rotation around its horizontal axle is denoted φ . The description therefore involves four constant parameters: α , l , r , and d , and two variables: $\varphi(t)$ and $\beta(t)$.

With these notations the kinematic constraints are derived as follows.

We make the derivation explicit for the general situation corresponding to a caster wheel (Fig. 17.2b). For fixed or steering wheels one just has to consider either the case $d = 0$ and constant β (fixed wheels), or $d = 0$ and variable β (steering wheels).

First we evaluate the velocity of the center of the wheel, which results from the following vector expression $\frac{d}{dt} \mathbf{OB} = \frac{d}{dt} \mathbf{OP} + \frac{d}{dt} \mathbf{PA} + \frac{d}{dt} \mathbf{AB}$. The two components of this vector in the robot frame are expressed as: $\dot{x} \cos \theta + \dot{y} \sin \theta - l \dot{\theta} \sin \alpha + (\dot{\theta} + \dot{\beta}) d \cos(\alpha + \beta)$ and $-\dot{x} \sin \theta + \dot{y} \cos \theta - l \dot{\theta} \cos \alpha + (\dot{\theta} + \dot{\beta}) d \sin(\alpha + \beta)$.

The projections of this vector onto the direction of the wheel plane, i.e., onto the vector $(\cos(\alpha + \beta - \pi/2), \sin(\alpha + \beta - \pi/2))$ and the vector of the wheel axle $(\cos(\alpha + \beta), \sin(\alpha + \beta))$, are $r\dot{\varphi}$ and 0, respectively, corresponding to the pure rolling and nonslip conditions.

After some manipulations, these conditions can be rewritten in the following compact form.

Pure rolling condition:

$$(-\sin(\alpha + \beta) \cos(\alpha + \beta) l \cos \beta) \mathbf{R}(\theta) \dot{\xi} + r \dot{\varphi} = 0, \quad (17.1)$$

Nonslip condition:

$$(-\cos(\alpha + \beta) \sin(\alpha + \beta) d + l \sin \beta) \mathbf{R}(\theta) \dot{\xi} + d \dot{\beta} = 0. \quad (17.2)$$

In these expressions $\mathbf{R}(\theta)$ is the orthogonal rotation matrix expressing the orientation of the robot with

respect to the inertial frame, i. e.,

$$\mathbf{R}(\theta) = \begin{pmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (17.3)$$

As said before, these general expressions can be simplified for the different types of conventional wheels.

For *fixed wheels*, the center of the wheel is fixed with respect to the cart and the wheel orientation is constant. This corresponds to a constant value of β and $d = 0$ (Fig. 17.2a). The nonslip equation (17.2) then reduces to

$$(\cos(\alpha + \beta) \quad \sin(\alpha + \beta) \quad l \sin \beta) \mathbf{R}(\theta) \dot{\xi} = 0. \quad (17.4)$$

For *steering wheels*, the center of the wheel is also fixed with respect to the cart (i. e., $d = 0$), with β time-varying, so the nonslip equation takes the form (17.2). This structure was already introduced in Fig. 17.2c.

The situation described by (17.1) and (17.2), with a nonzero-length rod AB and time-varying orientation angle β corresponds to *caster wheels*.

Swedish Wheels

The position of a Swedish wheel with respect to the cart is described, as for a fixed wheels, by three constant parameters: α , β , and l . An additional parameter is required to characterize the direction, with respect to the wheel plane, of the zero component of the velocity at the contact point of the wheel. This parameter is γ , which is the angle between the axle of the rollers and the wheel plane (Fig. 17.3b).

The kinematic constraints now impose only one condition:

$$\begin{pmatrix} -\sin(\alpha + \beta + \gamma) & \cos(\alpha + \beta + \gamma) & l \cos(\beta + \gamma) \end{pmatrix} \times \mathbf{R}(\theta) \dot{\xi} + r \cos \gamma \dot{\varphi} = 0. \quad (17.5)$$

17.2.3 Robot Configuration Variables

We now consider a wheeled robot equipped with N wheels of the above described types. We use the following subscripts to identify quantities related to these four types: 'f' for fixed wheels, 's' for steering wheels, 'c' for caster wheels, and 'sw' for Swedish wheels. The numbers of wheels of each type are denoted by N_f , N_s , N_c , and N_{sw} , with $N = N_f + N_s + N_c + N_{sw}$.

The configuration of the robot is fully described by the following generalized coordinate vector.

- *posture coordinates*: the posture vector $\xi(t) = (x(t) \ y(t) \ \theta(t))^T$;

- *orientation coordinates*: the $N_s + N_c$ orientation angles of the steering and caster wheels, i. e., $\beta(t) = (\beta_s(t) \ \beta_c(t))^T$;
- *rotation coordinates*: the N rotation angles of the wheels, i. e., $\varphi(t) = (\varphi_f(t) \ \varphi_s(t) \ \varphi_c(t) \ \varphi_{sw}(t))^T$

This whole set of coordinates is termed the set of configuration coordinates. The total number of configuration coordinates is $N_f + 2N_s + 2N_c + N_{sw} + 3$.

17.2.4 Restriction on Robot Mobility

The *pure rolling conditions* for fixed, steering, and caster wheels, as well as the constraints relative to the Swedish wheels, can be written in the following compact form

$$\mathbf{J}_1(\beta_s, \beta_c) \mathbf{R}(\theta) \dot{\xi} + \mathbf{J}_2 \dot{\varphi} = 0, \quad (17.6)$$

with

$$\mathbf{J}_1(\beta_s, \beta_c) = \begin{pmatrix} \mathbf{J}_{1f} \\ \mathbf{J}_{1s}(\beta_s) \\ \mathbf{J}_{1c}(\beta_c) \\ \mathbf{J}_{1sw} \end{pmatrix}.$$

In this expression \mathbf{J}_{1f} , $\mathbf{J}_{1s}(\beta_s)$, $\mathbf{J}_{1c}(\beta_c)$, and \mathbf{J}_{1sw} are, respectively, $(N_f \times 3)$, $(N_s \times 3)$, $(N_c \times 3)$, and $(N_{sw} \times 3)$ matrices, whose forms derive directly from the kinematic constraints, while \mathbf{J}_2 is a constant $(N \times N)$ diagonal matrix whose entries are the radii of the wheels, except for the radii of the Swedish wheels which are multiplied by $\cos \gamma$.

The value $\gamma = \frac{\pi}{2}$ would correspond to the direction of the zero component of the velocity being orthogonal to the plane of the Swedish wheel. Such a wheel would be subject to a constraint identical to the nonslip condition for a conventional wheel, hence losing the benefit of implementing a Swedish wheel. This implies that $\gamma \neq \frac{\pi}{2}$ and that \mathbf{J}_2 is a nonsingular matrix.

The *nonslip conditions* for caster wheels can be summarized as

$$\mathbf{C}_{1c}(\beta_c) \mathbf{R}(\theta) \dot{\xi} + \mathbf{C}_{2c} \dot{\beta}_c = 0, \quad (17.7)$$

where $\mathbf{C}_{1c}(\beta_c)$ is a $(N_c \times 3)$ matrix, whose entries derive from the nonslip constraints (17.2), while \mathbf{C}_{2c} is a constant diagonal nonsingular matrix, whose entries are equal to d .

The last constraints relate to the *nonslip conditions for fixed and steering wheels*. They can be summarized as

$$\mathbf{C}_1^*(\beta_s) \mathbf{R}(\theta) \dot{\xi} = 0, \quad (17.8)$$

where

$$\mathbf{C}_1^*(\beta_s) = \begin{pmatrix} \mathbf{C}_{1f} \\ \mathbf{C}_{1s}(\beta_s) \end{pmatrix},$$

where \mathbf{C}_{1f} and $\mathbf{C}_{1s}(\beta_s)$ are, respectively, $N_f \times 3$ and $N_s \times 3$ matrices.

It is important to point out that the restrictions on robot mobility result only from the conditions (17.8) involving the fixed and the steering wheels. These conditions imply that the vector $\mathbf{R}(\theta)\dot{\xi}$ belongs to $N[\mathbf{C}_1^*(\beta_s)]$, the null space of the matrix $\mathbf{C}_1^*(\beta_s)$. For any $\mathbf{R}(\theta)\dot{\xi}$ satisfying this condition, there exists a vector $\dot{\phi}$ and a vector $\dot{\beta}_c$ satisfying, respectively, conditions (17.6) and (17.7), because \mathbf{J}_2 and \mathbf{C}_{2c} are nonsingular matrices.

Obviously $\text{rank}[\mathbf{C}_1^*(\beta_s)] \leq 3$. If it is equal to 3 then $\mathbf{R}(\theta)\dot{\xi} = 0$, which means that any motion in the plane is impossible. More generally, restrictions on robot mobility are related to the rank of $\mathbf{C}_1^*(\beta_s)$, as will be discussed in detail below.

It is worth noticing that condition (17.8) has a direct geometrical interpretation. At each time instant the motion of the robot can be viewed as an instantaneous rotation about the instantaneous center of rotation (ICR), whose position with respect to the cart can be time varying. At each instant the velocity of any point of the cart is orthogonal to the straight line joining this point and the ICR. This is true, in particular, for the centers of the fixed and steering wheels, which are fixed points of the cart. On the other hand, the nonslip condition implies that the velocity of the wheel center is aligned with the wheel plane. These two facts imply that the horizontal rotation axes of the fixed and steering wheels intersect at the ICR (Fig. 17.5). This is equivalent to the condition that $\text{rank}[\mathbf{C}_1^*(\beta_s)] \leq 3$.

17.2.5 Characterization of Robot Mobility

As said before the mobility of the robot is directly related to the rank of $\mathbf{C}_1^*(\beta_s)$, which depends on the design of the robot. We define the degree of mobility δ_m as

$$\delta_m = 3 - \text{rank}[\mathbf{C}_1^*(\beta_s)]. \quad (17.9)$$

Let us first examine the case $\text{rank}(\mathbf{C}_{1f}) = 2$, which implies that the robot has at least two fixed wheels. If there are more than two fixed wheels, their axes intersect at the ICR, whose position with respect to the cart is then fixed in such a way that the only possible motion is a rotation of the cart about this fixed ICR. Obviously, from the user's point of view, such a design is not acceptable. We therefore assume that $\text{rank}(\mathbf{C}_{1f}) \leq 1$.

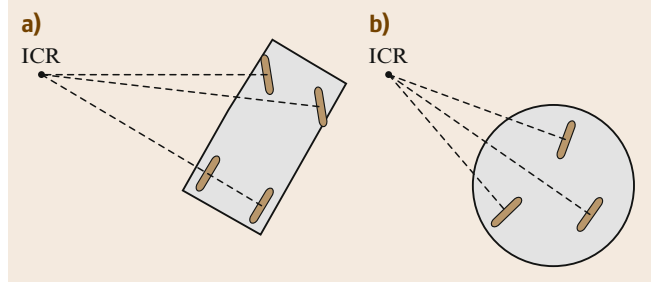


Fig. 17.5a,b The instantaneous center of rotation. (a) A car-like robot; (b) a three-steering-wheels robot

Moreover, we assume that $\text{rank}[\mathbf{C}_1^*(\beta_s)] = \text{rank}(\mathbf{C}_{1f}) + \text{rank}[\mathbf{C}_{1s}(\beta_s)] \leq 2$.

These two assumptions are equivalent to the following set of conditions.

1. If the robot has more than one fixed wheel, they are all on a single common axle.
2. The centers of the steering wheels do not belong to this common axle of the fixed wheels.
3. The number $\text{rank}[\mathbf{C}_1^*(\beta_s)]$ is equal to the number of steering wheels that can be oriented independently in order to steer the robot.

We call this number the *degree of steerability*:

$$\delta_s = \text{rank}[\mathbf{C}_{1s}(\beta_s)]. \quad (17.10)$$

If a robot is equipped with more than δ_s steering wheels, the motion of the extra wheels must be coordinated in order to guarantee the existence of the ICR at each instant.

We conclude that, for wheeled mobile robot of practical interest, the two defined indices, δ_m and δ_s , satisfy the following conditions.

1. The degree of mobility satisfies $1 \leq \delta_m \leq 3$. The upper bound is obvious, while the lower bound means that we consider only cases where motion is possible.
2. The degree of steerability satisfies $0 \leq \delta_s \leq 2$. The upper bound can be reached only for robots without fixed wheels, while the lower bound corresponds to robots without steering wheels.
3. The following is satisfied: $2 \leq \delta_m + \delta_s \leq 3$.

The case $\delta_m + \delta_s = 1$ is not acceptable because it corresponds to the rotation of the robot about a fixed ICR. The cases $\delta_m \geq 2$ and $\delta_s = 2$ are excluded because, according to the above assumptions, $\delta_s = 2$ implies $\delta_s = 1$. These conditions imply that only five structures are of practical interest, corresponding to the five pairs (δ_m, δ_s) satisfying the above inequalities, according to

the following array.

$$\begin{array}{cccccc} \delta_m & 3 & 2 & 2 & 1 & 1 \\ \delta_s & 0 & 0 & 1 & 1 & 2 \end{array}$$

Below, each type of structure will be designated by using a denomination of the form type (δ_m, δ_s) robot.

17.2.6 The Five Classes of Wheeled Mobile Robots

We now briefly describe the five classes of wheeled robot structures, pointing out the mobility restriction inherent to each class. Details and examples can be found in Sect. 17.5 and in [17.6].

Type (3,0) Robots

These robots have no fixed and no steering wheels and are equipped only with Swedish or caster wheels. Such robots are called *omnimobile*, because they have full mobility in the plane, which means that they are able to move in any direction without any reorientation.

Type (2,0) Robots

These robots have no steering wheels, but either one or several fixed wheels with a common axle. Mobility is restricted in the sense that, at a given posture $\xi(t)$, the velocity $\dot{\xi}(t)$ is constrained to belong to a two-dimensional distribution spanned by the vector fields $R^\top(\theta)s_1$ and $R^\top(\theta)s_2$, where s_1 and s_2 are two constant vectors spanning $N(C_{1f})$. A typical example of such a robot is the wheelchair.

Type (2,1) Robots

These robots have no fixed wheels and at least one steering wheel. If there is more than one steering wheel, their orientations must be coordinated in such a way that $\text{rank}[C_{1s}(\beta_s)] = \delta_s = 1$. The velocity $\dot{\xi}(t)$ is constrained to belong to a two-dimensional distribution spanned by the vector fields $R^\top(\theta)s_1(\beta_s)$ and $R^\top(\theta)s_2(\beta_s)$, where $s_1(\beta_s)$ and $s_2(\beta_s)$ are two vectors spanning $N(C_{1s}(\beta_s))$.

Type (1,1) Robots

These robots have one or several fixed wheels on a single common axle, and also one or several steering wheels, with the conditions that their centers are not located on the common axle of the fixed wheels, and that their orientations are coordinated. The velocity $\dot{\xi}(t)$ is constrained to belong to a one-dimensional distribution parameterized by the orientation angle of one arbitrarily chosen steering wheel. Mobile robots built on the model of a conventional car (often called *car-like* robots) belong to this class.

Type (1,2) Robots

These robots have no fixed wheels, but at least two steering wheels. If there are more than two steering wheels, then their orientation must be coordinated in order to satisfy the condition $\text{rank}[C_{1s}(\beta_s)] = \delta_s = 2$. The velocity $\dot{\xi}(t)$ is constrained to belong to a one-dimensional distribution parameterized by the orientation angles of two arbitrarily chosen steering wheels.

17.3 State-Space Models of Wheeled Mobile Robots

In this section, the mobility analysis discussed in the previous section is reformulated into a state-space form that will be useful for subsequent developments. We introduce four different kinds of state-space representation that are of interest for understanding the behavior of wheeled robots, and for control design purpose.

- The *posture kinematic model*, which is the simplest state-space model able to give a global description of the robot, from the users viewpoint.
- The *configuration kinematic model* allows one to describe the kinematic behavior of the whole robot, including all the configuration variables.
- The *configuration dynamic model* is the most general state-space model. It gives a complete description of

the dynamics including the forces provided by the actuators. In particular, it allows to one address the issue of actuator configuration and to define a criterion to check whether the motorization is sufficient to exploit the kinematic mobility fully.

- The *posture dynamic model*, which is feedback equivalent to the configuration dynamic model, constitutes a dynamical counterpart to the posture kinematic model.

17.3.1 Posture Kinematic Models

We have shown that, whatever the type of robot, the velocity vector $\dot{\xi}(t)$ is restricted to belong to a distribution

Δ_c defined as

$$\dot{\xi} \in \Delta_c = \text{span}\{\text{col}[\mathbf{R}^\top(\theta)\Sigma(\beta_s)]\},$$

where the columns of the matrix $\Sigma(\beta_s)$ constitute a basis of $N[\mathbf{C}_1^*(\beta_s)]$. This is equivalent to the following statement: for all t , there exists a vector η such that

$$\dot{\xi} = \mathbf{R}^\top(\theta)\Sigma(\beta_s)\eta, \quad (17.11)$$

The dimension of the distribution Δ_c , and hence of the vector $\eta(t)$, is equal to the degree of mobility δ_m of the robot. Obviously, in the case where the robot has no steering wheels, the matrix Σ is constant, and the expression (17.11) reduces to

$$\dot{\xi} = \mathbf{R}^\top(\theta)\Sigma\eta. \quad (17.12)$$

In the opposite case ($\delta_s \geq 1$), the matrix Σ explicitly depends on the orientation angles β_s , and the expression (17.11) can be augmented as follows:

$$\dot{\xi} = \mathbf{R}^\top(\theta)\Sigma(\beta_s)\eta, \quad (17.13)$$

$$\dot{\beta}_s = \zeta. \quad (17.14)$$

The representation (17.12) (or (17.13) and (17.14)) can be viewed as a state-space representation of the model, reflecting the mobility restriction induced by the constraints; it is termed the *posture kinematic model*.

The state vector is constituted by the three posture coordinates $\dot{\xi}(t)$ and, possibly, by δ_s orientation coordinates β_s . The vectors η and ξ , of dimension δ_m and δ_s , respectively, are homogeneous to velocities and can be interpreted as control inputs entering the model linearly.

Nevertheless, this interpretation should be treated with some care, since the true physical inputs are the torques provided by the embarked actuators. The kinematic posture model is in fact only a subsystem of the general dynamic model that will be presented in Sect. 17.3.3.

This posture kinematic model allows us to discuss further the maneuverability of wheeled robots. The degree of mobility δ_m is equal to the number of degrees of freedom that can be *directly* manipulated from the inputs $\eta(t)$, without reorientation of the steering wheels. Intuitively, it corresponds to how many degrees of freedom the robot could have instantaneously from its current position, without steering any of its wheels. This number δ_m is not equal to the overall number of degrees of freedom of the robot that can be manipulated from the inputs $\eta(t)$ and $\zeta(t)$, which is equal to the sum $\delta_M = \delta_m + \delta_s$ and which we could call degree of maneuverability. It includes the δ_s degrees of freedom that are accessible from

the inputs $\zeta(t)$. However, the action of $\zeta(t)$ on the posture coordinates $\dot{\xi}(t)$ is indirect, since it is achieved only through the coordinates β_s , which are related to the inputs $\zeta(t)$ by an integral action, reflecting the fact that the modification of the orientation of a steering wheel cannot be achieved instantaneously. The maneuverability of a wheeled robot depends not only on δ_M , but also on the way these δ_M degrees of freedom are partitioned into δ_m and δ_s . Therefore, two indices are needed to characterize the maneuverability. Obviously the ideal situation is that of omnimobile robots where $\delta_M = \delta_m = 3$.

In order to avoid useless notational complications, we will assume from now on that the degree of steerability is equal to the number of steering wheels, i.e., $N_s = \delta_s$. This is a restriction from a robot design viewpoint. However, for the mathematical analysis of the behavior of mobile robots, there is no loss of generality in this assumption, although it considerably simplifies the technical derivation. Indeed, for robots with an excess of steering wheels, it is always possible to reduce the condition (17.8) to a minimal subset of exactly δ_s independent constraints that correspond to the δ_s wheels that have been selected as the master steering wheels and to ignore the other slave wheels in the analysis.

17.3.2 Configuration Kinematic Models

In order to discuss the restriction of mobility, we have considered only a subset of the conditions induced by the kinematic constraints, namely the nonslip conditions for the fixed and steering wheels (17.8). The remaining constraints are now used to derive the equations of evolution of the rotation velocities $\dot{\varphi}$ and of the orientation velocities of the caster wheels $\dot{\beta}_c$. From (17.6) and (17.7) it follows immediately that

$$\dot{\beta}_c = -\mathbf{C}_{2c}^{-1}\mathbf{C}_{1c}(\beta_c)\mathbf{R}(\theta)\dot{\xi}, \quad (17.15)$$

$$\dot{\varphi} = -\mathbf{J}_2^{-1}\mathbf{J}_1(\beta_s, \beta_c)\mathbf{R}(\theta)\dot{\xi}. \quad (17.16)$$

By combining these equations with the posture kinematic model (17.13), the state equations for β_c and φ become

$$\dot{\beta}_c = -\mathbf{D}(\beta_c)\Sigma(\beta_s)\eta, \quad (17.17)$$

$$\dot{\varphi} = -\mathbf{E}(\beta_s, \beta_c)\Sigma(\beta_s)\eta, \quad (17.18)$$

where

$$\mathbf{D}(\beta_c) = -\mathbf{C}_{2c}^{-1}\mathbf{C}_{1c}(\beta_c) \quad \text{and}$$

$$\mathbf{E}(\beta_s, \beta_c) = -\mathbf{J}_2^{-1}\mathbf{J}_1(\beta_s, \beta_c).$$

Defining q as the vector of configuration coordinates, i. e.,

$$q = \begin{pmatrix} \xi \\ \beta_s \\ \beta_c \\ \varphi \end{pmatrix},$$

the evolution of the configuration coordinates can be described by the following compact form, called the *configuration kinematic model*

$$\dot{q} = S(q)u, \quad (17.19)$$

where

$$S(q) = \begin{pmatrix} R^\top(\theta)\Sigma(\beta_s) & 0 \\ 0 & I \\ D(\beta_c)\Sigma(\beta_s) & 0 \\ E(\beta_s, \beta_c)\Sigma(\beta_s) & 0 \end{pmatrix},$$

and $u = \begin{pmatrix} \eta \\ \varsigma \end{pmatrix}.$ (17.20)

The vector q is the vector of generalized coordinates allowing one to fully describe the position and the configuration of the mobile robot.

The constraints (17.6), (17.7), and (17.8) can be summarized under the following compact form

$$J(q)\dot{q} = 0, \quad (17.21)$$

where the matrix $J(q)$ is the Jacobian of the constraints, i. e.,

$$J(q) = \begin{pmatrix} J_1(\beta_s, \beta_c)R(\theta) & 0 & 0 & J_2 \\ C_{1c}(\beta_c)R(\theta) & 0 & C_{2c} & 0 \\ C_1^*(\beta_s)R(\theta) & 0 & 0 & 0 \end{pmatrix}. \quad (17.22)$$

The two matrices $S(q)$ and $J(q)$ satisfy the relation

$$S(q)J(q) = 0. \quad (17.23)$$

17.3.3 Configuration Dynamic Models

We now derive the dynamic configuration model which constitutes, from a mechanical viewpoint, the full system description, allowing one to relate the control inputs provided by the embarked actuators to the evolution of the generalized coordinates q . This model is derived using the Lagrange formalism.

We assume that the robot is equipped with actuators that can force either the orientation of the steering and

caster wheels, or the orientation of all the wheels. The torques provided by the actuators are denoted τ_φ for the rotation of the wheels, τ_c for the orientation of the caster wheels, and τ_s for the orientation of the steering wheels.

Using the Lagrange formalism, we obtain the following compact equations:

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}} \right) - \frac{\partial T}{\partial q} = \tau + J^\top(q)\lambda. \quad (17.24)$$

In this equation

1. $T(q, \dot{q})$ the kinetic energy of the robot can be expressed as a quadratic form $T(q, \dot{q}) = \frac{1}{2}\dot{q}^\top M(\beta_c)\dot{q}$, where $M(\beta_c)$ is a symmetric positive-definite matrix.
2. τ is the vector of the generalized forces associated with the torques provided by the actuators

$$\tau = \begin{pmatrix} 0 \\ \tau_s \\ \tau_c \\ \tau_\varphi \end{pmatrix},$$

3. the term $J^\top(q)\lambda$ is the vector of generalized forces associated with the kinematic constraints. The vector λ is the vector of the Lagrange multipliers associated with the constraints.

With the above expression for the kinetic energy (17.19) can be rewritten as

$$M(\beta_c)\ddot{q} + f(q, \dot{q}) = \tau + J^\top(q)\lambda. \quad (17.25)$$

(See Sect. 2.3 on the dynamics of rigid-body systems.)

The Lagrange multipliers are eliminated by left multiplying (17.25) by $S(q)$ and using (17.23). Moreover, \ddot{q} and \dot{q} can be eliminated using (17.19).

It follows that

$$\begin{aligned} & [S^\top(q)M(\beta_c)S(q)]\ddot{u} \\ & + [S^\top(q)M(\beta_c)\dot{S}(q)u + S^\top(q)f(q, S(q)u)] \\ & = S^\top(q)\tau \end{aligned} \quad (17.26)$$

or, more compactly

$$H(q)\ddot{u} + F(q, u) = S^\top(q)\tau. \quad (17.27)$$

This equation, together with the equation of the configuration kinematic model (17.19) constitutes the configuration dynamic model of the robot.

In this general formulation (17.27) τ represents all the torques that can potentially be applied for the orientation and rotation of the wheels. In practice, however, only

a limited number of actuators are used, which means that many components of this vector are identically zero. Our purpose now is to characterize the actuator configurations exploiting the maneuverability that can be expected from the posture kinematic model.

First, it is clear that all the steering wheels must be provided with an actuator for their orientation, otherwise these wheels would just play the role of fixed wheels.

Moreover, in order to ensure full robot mobility, N_m additional actuators (with $N_m \geq \delta_m$) must be implemented for either the rotation of some wheels or the orientation of some caster wheels. The vector of the torques developed by these additional actuators is denoted by τ_m , and we can write

$$\begin{pmatrix} \tau_c \\ \tau_\varphi \end{pmatrix} = \mathbf{P} \tau_m, \quad (17.28)$$

where \mathbf{P} is an $((N_c + N) \times N_m)$ elementary matrix that selects the components of τ_c and τ_φ that appear explicitly in τ_m and may therefore be used as control inputs. With this notation, the dynamic equation (17.27) can be rewritten as

$$\begin{aligned} \mathbf{H}(q)\dot{\mathbf{u}} + \mathbf{F}(q, u) &= \begin{pmatrix} \mathbf{I} & 0 \\ 0 & \mathbf{B}(\beta_s, \beta_c) \mathbf{P} \end{pmatrix} \begin{pmatrix} \tau_s \\ \tau_m \end{pmatrix} \\ &= \mathbf{\Gamma}(\beta_s, \beta_c) \begin{pmatrix} \tau_s \\ \tau_m \end{pmatrix}, \end{aligned} \quad (17.29)$$

where $\mathbf{B}(\beta_s, \beta_c) = \mathbf{\Sigma}^\top(\beta_c)(\mathbf{D}^\top(\beta_c) \mathbf{E}^\top(\beta_s, \beta_c))$

It must be pointed out that the evolution of the variable u can be assigned provided \mathbf{P} (and equivalently the actuator implementation) is such that the matrix $\mathbf{B}(\beta_s, \beta_c)\mathbf{P}$ has full rank for all (β_s, β_c) . We assume in the sequel that this condition is satisfied, ensuring therefore full exploitation of the potential maneuverability of the robot.

17.3.4 Posture Dynamic Models

It follows from the above assumption that the configuration dynamic model is feedback equivalent (by smooth static time-invariant state feedback) to the following system:

$$\dot{q} = S(q)u, \quad (17.30)$$

$$\dot{\mathbf{u}} = \mathbf{v}, \quad (17.31)$$

where \mathbf{v} represents a set of $(\delta_s + \delta_m)$ auxiliary independent control inputs.

This state feedback is given by

$$\begin{pmatrix} \tau_s \\ \tau_m \end{pmatrix} = \mathbf{\Gamma}^+(\beta_s, \beta_c) [\mathbf{H}(q)v - \mathbf{F}(q, u)], \quad (17.32)$$

where $\mathbf{\Gamma}^+$ denotes an arbitrary left inverse of $\mathbf{\Gamma}$.

(See Sect. 6.6 on computer torque control.)

We emphasize that a further simplification is of interest from an operational viewpoint. In the context of trajectory planning or feedback control design, it is clear that the user is essentially concerned with controlling the posture of the robot (namely the posture coordinate $\xi(t)$) by using the control input v . This implies that we can deliberately ignore the coordinates β_c and φ and restrict our attention to the following posture dynamic model:

$$\dot{z} = \mathbf{B}(z)u, \quad (17.33)$$

$$\dot{\mathbf{u}} = \mathbf{v}, \quad (17.34)$$

where

$$\begin{aligned} z &= \begin{pmatrix} \xi \\ \beta_s \end{pmatrix}, \quad u = \begin{pmatrix} \eta \\ s \end{pmatrix} \text{ and} \\ \mathbf{B}(z) &= \begin{pmatrix} R^\top(\theta) \mathbf{\Sigma}(\beta_s) & 0 \\ 0 & \mathbf{I} \end{pmatrix}. \end{aligned}$$

The first equation is nothing but the kinematic posture model equation (17.12), or (17.13, 17.14). The difference with this kinematic model is the presence of integrators on the input variables (17.34), so that the variables u are now part of the state vector. This leads to the appearance of a drift term in the dynamic model.

The posture dynamic model fully describes the system dynamics between the control input v and the posture ξ . The coordinates β_c and φ have apparently disappeared but it is important to note that they are in fact hidden in the feedback (17.32).

17.3.5 Articulated Robots

Up to now we have considered only simple mobile robots, i.e., robots made up of only one cart. In this section we extend this analysis to articulated robots, i.e., robots formed by a master cart with several trailers. A typical example is the well-known truck and trailers system.

Each of the trailers can be equipped of wheels of the types described in Sect. 17.2.2, actuated or not. The number of possible structures becomes almost infinite. This is why we restrict our analysis to passive trailers (no onboard actuators) equipped only with fixed wheels.

We consider a master cart that can be described by any of the five discussed types. The first trailer is connected to the cart, i.e., a material point of the trailer is fixed to a material point of the master cart. The relative position of the trailer with respect to the cart is described by a posture angle θ_1 . This trailer is equipped with one or several fixed wheels. If there is more than one wheel, then they have the same geometric axle. To this first trailer is connected a second trailer, of the same type, whose relative position with respect to the first is characterized by the angle θ_2 , and so on, for a sequence of N_t trailers.

The posture vector allowing one to describe this augmented system is now extended to a $3 + N_t$ generalized posture vector: $\xi^* = \begin{pmatrix} \xi \\ \theta^* \end{pmatrix}$, where ξ is the posture vector of the master cart while θ^* is the vector of the relative angles of the successive trailers. We also denote by ξ_i^* the posture of a partial system, from the master cart up to the i -th trailer, i.e.,

$$\xi_i^* = \begin{pmatrix} \xi^\top & \theta_1 & \dots & \theta_i \end{pmatrix}^\top.$$

The kinematic constraints relative to the fixed wheel j of trailer i ($1 \leq i \leq N_t$) can be expressed as in Sect. 17.2.2 using the following notations:

- The reference point is the connection point between trailer $(i - 1)$ and trailer i .
- We define the posture of trailer i , ξ_i , by the vector made up by the coordinates of this reference point and the absolute orientation of the trailer, which is equal to $(\theta + \theta_1 + \dots + \theta_i)$.
- The position of the wheel with respect to this reference point is described by its polar coordinates α_{ij} and l_{ij} . The orientation of the wheel is given by the constant angle β_{ij} .
- The rotation angle of the wheel is denoted by φ_{ij} .

The *pure rolling condition* is then given by

$$(-\sin(\alpha_{ij} + \beta_{ij}) \cos(\alpha_{ij} + \beta_{ij}) l_{ij} \cos \beta_{ij} + R(\theta + \theta_1 + \dots + \theta_i) \dot{\xi}_i + r \dot{\varphi}_{ij}) = 0, \quad (17.35)$$

while the *nonslip condition* becomes

$$\cos(\alpha_{ij} + \beta_{ij}) \sin(\alpha_{ij} + \beta_{ij}) l_{ij} \sin \beta_{ij} + R(\theta + \theta_1 + \dots + \theta_i) \dot{\xi}_i = 0. \quad (17.36)$$

The posture vector of trailer i , ξ_i , can be expressed as a function of ξ and of the orientation angles of the preceding trailers, i.e.,

$$\xi_i = g_i(\xi_{i-1}^*).$$

This implies that

$$\begin{aligned} \dot{\xi}_i &= \frac{\partial g_i}{\partial \xi} \dot{\xi} + \sum_{k=1}^{i-1} \frac{\partial g_i}{\partial \theta_k} \dot{\theta}_k \\ &= G_i(\xi_{i-1}^*) \dot{\xi}_{i-1}^*, \quad i = 1, \dots, N_t. \end{aligned} \quad (17.37)$$

Using this expression in the nonslip condition (17.36) for the N_t trailers, and using the nonslip condition for the master cart, we obtain the following set of equations reflecting the restriction of motion for all parts of the system:

$$J^*(\beta_s, \theta, \theta_1, \dots, \theta_{N_t}) \dot{\xi}^* = 0. \quad (17.38)$$

Following the same lines as in Sect. 17.3.1, we derive the *posture kinematic model* of the articulated robot:

$$\dot{\xi}^* = S^*(\beta_s, \theta, \theta_1, \dots, \theta_{N_t}) \eta, \quad (17.39)$$

$$\dot{\beta}_s = \varsigma. \quad (17.40)$$

The matrices J^* and S^* satisfy the relation

$$J^*(\beta_s, \theta, \theta_1, \dots, \theta_{N_t}) S^*(\beta_s, \theta, \theta_1, \dots, \theta_{N_t}) = 0. \quad (17.41)$$

Equation (17.37) provides the evolution of the relative angles of each trailer. It must be noticed that the time derivative of each of these angles depends on the input variables relative to the master cart (i.e., η and ς), on β_s , if the master cart is equipped with steering wheels, and on the relative angles of the preceding trailers only. This last property is due to the recursive structure of J^* and S^* .

Obviously it is possible to derive the three other models of the articulated robot, in a similar way as for the simple robot, as explained in Sects. 17.3.2, 17.3.3, and 17.3.4.

For the *configuration kinematic model* we consider the generalized vector constituted from all the variables, including the rotation angles of the wheels of the trailers. This model is derived as an extension of the posture model taking into account the pure rolling constraints (17.35). It has the form (17.19), with an appropriate definition of q . The *configuration dynamic model* is derived from the Lagrange equations of the system. It has the form (17.27), while the *posture dynamic model*, obtained via static state feedback, is again related to the posture kinematic, as for the simple robot (17.33) and (17.34).

17.4 Structural Properties of Wheeled Robots Models

Our purpose in this section is to discuss the structural properties of the above models of wheeled robots from a control design viewpoint. Since, in most situations, the user is only interested in the posture of the robot, and not in the *internal* variables (such as the wheel orientation angles), the most interesting models are the posture models (kinematic or dynamic). This is why the discussion on structural properties will be mainly based on the posture models.

17.4.1 Irreducibility, Controllability, and Nonholonomy

1. We first address the question of the reducibility of the *kinematic posture* state-space model (17.33). A state model is reducible if there exists a change of coordinates such that some of the new coordinates are identically zero along the motion system. For a nonlinear dynamical system without drift like (17.13, 14) reducibility is related to the dimension of the involutive closure $\bar{\Delta}$ of the following distribution Δ , expressed in local coordinates as

$$\Delta(z) = \text{span} [\text{col} B(z)] .$$

A well-known consequence of the Frobenius theorem is that the system is reducible only if $\dim(\bar{\Delta}) \leq \dim(\Delta) - 1$.

The following property can be checked for the *posture kinematic models* of wheeled robots.

- For the posture kinematic model (17.33) $\dot{z} = B(z)u$,
- the input matrix $B(z)$ has full rank, i.e., $\text{rank} [B(z)] = \delta_m + \delta_s \forall z$,
 - the involutive distribution $\bar{\Delta}(z)$ has constant maximal dimension, i.e., $\dim [\bar{\Delta}(z)] = 3 + \delta_s$.

As a consequence, the posture kinematic model of a wheeled robot is *irreducible*. This is a coordinate-free property.

This property has another consequence related to the controllability of the posture kinematic model. For a nonlinear dynamical model without drift of the form (17.13, 14), the strong accessibility algebra coincides with the involutive distribution $\bar{\Delta}(z)$, which has constant maximal dimension. It follows that the strong accessibility rank condition is satisfied and, therefore, the system is strongly accessible from any configuration. For such a driftless system this implies *controllability*. Practically, this means that a mobile robot can always be driven from any initial

posture ξ_0 to any final one ξ_f , in a finite time, by manipulating the velocity control inputs $u = (\eta^\top \zeta^\top)^\top$. Finally, the difference between the dimensions of the two distributions $\Delta(z)$ and $\bar{\Delta}(z)$, i.e.,

$$\begin{aligned} \dim [\bar{\Delta}(z)] - \dim [\Delta(z)] &= (3 + \delta_s) - (\delta_m + \delta_s) \\ &= 3 - \delta_m , \end{aligned}$$

is related to the nonholonomy of the posture kinematic model.

If this difference is nonzero (i.e., if $\delta_m \leq 2$) the posture kinematic model is said to be *nonholonomic*. If $\delta_m = 3$, which is the case only for omnimobile robots, the kinematic posture model is *holonomic*.

2. The *configuration kinematic model* (17.23) is obtained from the posture model by adding the evolution of the internal variables $\beta_c(t)$ and $\varphi(t)$, and takes the same form $\dot{q} = S(q)u$.

In order to analyze reducibility and controllability issues we now have to consider the following two distributions: $\Delta_1(q) = \text{span} [\text{col}(S(q))]$, and its involutive closure $\bar{\Delta}_1(q)$.

It follows immediately that

$$\begin{aligned} \delta_m + N_s &= \dim [\Delta_1(q)] \leq \dim [\bar{\Delta}_1(q)] \leq \dim(q) \\ &= 3 + N + N_c + N_s . \end{aligned}$$

We define the *degree of nonholonomy* of the configuration kinematic model as

$$M = \dim [\bar{\Delta}_1(q)] - (\delta_m + \delta_s) .$$

This number represents the number of velocity constraints that are not integrable and therefore cannot be eliminated from the configuration evolution description, whatever the choice of the generalized coordinates. It must be pointed out that this number depends on the particular structure of the robot, and thus it has not necessarily the same value for two robots belonging to the same class.

On the other hand, for a particular choice of generalized coordinates, the number of coordinates that can be eliminated by integration of the constraints is equal to the difference between $\dim(q)$ and $\dim(\Delta_1(q))$.

It can be checked that the *configuration kinematic* model of all types of wheeled robots (including omnimobile robots) is *nonholonomic* (i.e., the degree of nonholonomy is not equal to zero), but is *reducible*.

Moreover, it does not satisfy the strong accessibility rank condition.

This property does not contradict the irreducibility of the posture kinematic model. The reducibility of the configuration model means that there exists at least one smooth function of $q(t)$, involving explicitly at least one of the variables $\beta_c(t)$ and $\varphi(t)$, that is constant along the trajectories of the system compatible with the full set of kinematic constraints (17.21).

3. We have seen in Sects. 17.3.3 and 17.3.4 that the dynamic models ((17.30, 31), for the configuration model, or (17.33, 34) for the posture model) are related to the corresponding kinematic models, with the difference that the variables are part of the state vector. This implies the existence of a drift term and the fact that the input vector fields are constant. The dynamic models inherit the structural properties of the corresponding dynamic model. In particular, the posture dynamic model is irreducible and small-time locally controllable.

17.4.2 Stabilizability

Obviously, any arbitrary configuration $(\bar{\xi}^\top \bar{\beta}_s^\top)^\top$ may constitute an equilibrium point for the posture kinematic model. Equilibrium means that the robot is at rest somewhere, with a given constant posture $\bar{\xi}$ and a given constant orientation $\bar{\beta}_s$ of the steering wheels, and zero velocities, i. e., $\bar{u} = 0$.

Let us now consider the question of the existence of a feedback control $u(z)$ able to stabilize a mobile robot at a particular \bar{z} .

For *holonomic robots* (i. e., omnimobile robots, for which z reduces to ξ) the question is trivial, because $\dot{\xi}$ can be assigned arbitrarily from the input u . For instance the state-feedback control $u(z) = B^{-1}(z)A(z - \bar{z})$, ensures the following closed-loop dynamics

$$\dot{\xi} = A(\xi - \bar{\xi}),$$

and therefore, for any arbitrary Hurwitz matrix A , the exponential stability of the equilibrium \bar{z} .

For restricted mobility robots (*nonholonomic robots*) the situation is less favorable. Indeed the so-called Brockett necessary condition for the existence of smooth time-invariant stabilizing feedback is not satisfied, since the map $(z, u) \rightarrow B(z)u$ is not onto on a neighborhood of the equilibrium \bar{z} (Sect. 34.4.4).

We conclude that for restricted mobility robots the (nonholonomic) *posture kinematic* model is not stabilizable by a continuous static time-invariant state feedback.

Nevertheless such a model can be stabilized using smooth time-varying state feedback or noncontinuous feedback.

This property is inherited by the corresponding *dynamic configuration model*, which is also not stabilizable by continuous static time-invariant state feedback.

17.4.3 Static State-Feedback Linearizability

In this section we analyze the existence of state feedback achieving full or partial linearization of the posture models given by (17.33) for the kinematic model, or (17.33, 34) for the dynamic model.

A first result is the following. The largest subsystem of the posture kinematic model (17.33) linearizable by a smooth static feedback has dimension $(\delta_m + \delta_s)$. The largest linearizable subsystem of the dynamic model (17.33, 34) has dimension $2(\delta_m + \delta_s)$.

It can be checked that the largest linearizable subsystem of the kinematic posture model is obtained by selecting $(\delta_m + \delta_s)$ adequate linearizing output functions depending on z . A vector of $(3 - \delta_m)$ components remains nonlinearized. Similarly the largest linearizable subsystem of the posture dynamic model has dimension $2(\delta_m + \delta_s)$, with exactly the same linearizing output functions.

For omnimobile robots (i. e., robots with holonomic posture models), $\delta_m = 3$ and $\delta_s = 0$. The above property implies that the posture models are fully linearizable by static state feedback. In contrast, restricted mobility robots posture models are only partially linearizable.

17.4.4 Dynamic State-Feedback Linearizability – Differential Flatness

The purpose of this section is to show that posture models of wheeled robots can be fully linearized by dynamic state feedback. Obviously we only consider restricted mobility robots, because omnidirectional robots are fully linearizable by static state feedback, as discussed in the previous section.

Consider a nonlinear dynamical system given in general state-space form, affine in the inputs

$$\dot{z} = f(z) + \sum_{i=1}^m g_i(z)u_i, \quad (17.42)$$

where the state $z \in R^n$, the input $u \in R^m$, and the vector fields f and g_i are smooth.

When the system is not completely linearizable by diffeomorphism and static state feedback (as for

restricted mobility robots), full linearization can nevertheless possibly be achieved by considering more general dynamic feedback laws of the form

$$\begin{aligned} u &= \alpha(z, \chi, w), \\ \dot{\chi} &= a(z, \chi, w), \end{aligned} \quad (17.43)$$

where w is an auxiliary control input. Such a dynamic feedback is obtained through the choice of m suitable linearizing output functions

$$y_i = h_i(z), \quad i = 1, \dots, m. \quad (17.44)$$

We apply the so-called *dynamic extension algorithm* to system (17.42)–(17.44). The idea of this algorithm is to delay some *combinations of inputs* simultaneously affecting several outputs, via the addition of integrators, in order to enable other inputs to act in the meanwhile and therefore hopefully to extend an extended decoupled system of the form

$$y_k^{(r_k)} = w_k, \quad k = 1, \dots, m, \quad (17.45)$$

where $y_k^{(i)}$ denotes the i -th derivative of y_k with respect to time, r_k is the relative degree of y_k , and w_k denote the new auxiliary inputs. In order to get a *full* linearization, we shall have for the n_e -dimensional extended system

$$\sum_{i=1}^m r_i = n_e, \quad (17.46)$$

17.5 Wheeled Robot Structures

There are many design alternatives for wheeled mobile robots. Design problems of a single-body mobile robot include the selection of wheel types, the placement of wheels, and the determination of the kinematic parameters. Design objectives should be specified according to the target environments and tasks, as well as the initial and operational costs of a robot. In this section, robot structures are classified according to the number of wheels, and then features will be introduced focusing on commonly adopted designs.

17.5.1 Robots with One Wheel

A robot with a single wheel is basically unstable without dynamic control in order to maintain its balance of a body. A typical example is a unicycle. As a variation of a unicycle, a robot with a rugby-ball-shaped wheel

where n_e is the dimension of the extended state vector $z_e = (z^\top \chi^\top)^\top$. If the condition (17.46) is satisfied, then

$$\zeta = \Psi(z_e) = (y_1 \dots y_1^{(r_1-1)} \dots y_m \dots y_m^{(r_m-1)})^\top$$

is a local diffeomorphism.

A system that can be shown to be linearizable by using the dynamic extension algorithm is an example of a so-called *differentially flat system*. The linearizing outputs $y_i = h_i(z)$ are also called flat outputs. Conversely, differential flatness refers by definition to a system that can be linearized by dynamic state feedback. Roughly speaking, for a flat system, the state and the inputs can be expressed as algebraic functions of the flat outputs and their successive time derivatives.

These two related properties of linearizability by dynamic state feedback and differential flatness will be of interest for control design and path planning problems.

The fact that the posture kinematic and dynamic models of restricted mobility robots are differentially flat systems result directly from the following two properties.

1. Any controllable driftless system with m inputs and at most $m + 2$ states is a differentially flat system.
2. If a nonlinear system $\dot{z} = f(z, u)$ is differentially flat, then the augmented system $\dot{z} = f(z, u)$, $\dot{u} = v$ is also a differentially flat system and is therefore generically fully linearizable by dynamic state feedback.

can be used in order to improve stability in the lateral direction, as studied in [17.7].

A spherical robot can also be considered as a single-wheel robot. A balancing mechanism such as a spinning wheel is employed to achieve dynamic stability. This approach has advantages including high maneuverability and low rolling resistance. However, single-wheel robots are rarely used in practical applications, because additional balancing mechanisms are required, control is difficult, and pose estimation by pure dead reckoning is not available. An example of a spherical robot can be found in [17.8].

17.5.2 Robots with Two Wheels

In general, there are two types of two-wheel robots, as shown in Fig. 17.6. Figure 17.6a shows a bicycle-

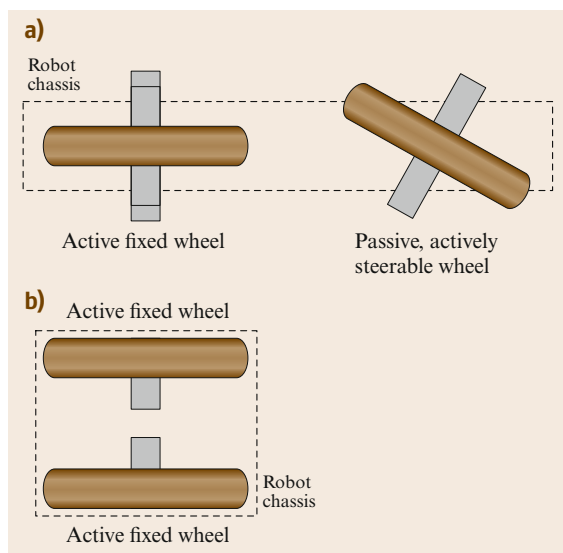


Fig. 17.6 (a) Bicycle-type robot, (b) inverted-pendulum-type robot

type robot. It is common to steer a front wheel and to drive a rear wheel. Since the dynamic stability of a bicycle-type robot increases with its speed, a balancing mechanism is not necessarily required. The advantage of this approach is that the robot width can be reduced. However, a bicycle type is rarely used because it cannot maintain its pose when the robot stands still. Fig. 17.6b shows an inverted-pendulum-type robot. It is a two-wheel differential drive robot.

It is possible to achieve static stability by accurately placing the center of gravity on the wheel axle. However, it is common to apply dynamic balancing control, which is similar to the conventional control problem for an inverted pendulum. The size of a robot can be reduced by using two-wheel robots, when compared with robots with more than three wheels. A typical application of a pendulum-type robot is to design a structure as a four-wheel robot, consisting of two pendulum robots connected. Then, the robot can climb stairs by lifting its front wheels while the robot reaches the stair. A major disadvantage is that control effort is always required for dynamic balancing. Examples of inverted-pendulum-type robots can be found in [17.9] and [17.10].

17.5.3 Robots with Three Wheels

Since a robot with three wheels is statically stable and has a simple structure, it is one of the most widely used

structures for wheeled robots. There are a large number of designs according to the choice of individual wheel types. Every wheel introduced in Sect. 17.2.1 can be used to construct three-wheel robots. In this section, five popular design examples are described: (1) two-wheel differential drive, (2) synchronous drive, (3) an omnimobile robot with Swedish wheels, (4) an omnimobile robot with active caster wheels, and (5) an omnidirectional robot with steerable wheels.

Two-Wheel Differential-Drive Robot

A two-wheel differential-drive robot is one of the most popular designs and is composed of two active fixed wheels and one passive caster wheel. The robot can be classified as a type (2,0) robot in the nomenclature of Sect. 17.2.6. It is possible to extend the robot to a four-wheel robot by adding passive caster wheels. The major advantages of the robot can be summarized as follows:

- A simple mechanical structure, a simple kinematic model, and low fabrication cost.
- A zero turning radius is available. For a cylindrical robot, the obstacle-free space can easily be computed by expanding obstacle boundaries by the robot radius r .
- Systematic errors are easy to calibrate.

On the other hand, its drawbacks are:

- difficulty of moving irregular surfaces. When the robot goes over uneven surfaces, its orientation might change abruptly if one of the active wheels loses contact with the ground;
- only bidirectional movement is available.

Synchronous-Drive Robot

A synchronous-drive robot can be built by using centered or off-centered orientable wheels. The steering and driving motions of each wheel are mechanically coupled by chains or belts, and the motions are actuated synchronously, so the wheel orientations are always identical. The kinematic model of a synchronous drive robot is equivalent to that of the unicycle, a type (1,1) robot. Therefore, omnidirectional motion, i. e., motion in any direction, can be achieved by steering the wheel orientations to the desired velocity direction. However, the orientation of the robot chassis cannot be changed. Sometimes a turret is employed to change the body orientation. The most significant advantage of the synchronous-drive robot is that omnidirectional movement can be achieved by using only two actuators. Since the mechanical structure guarantees synchronous steer-

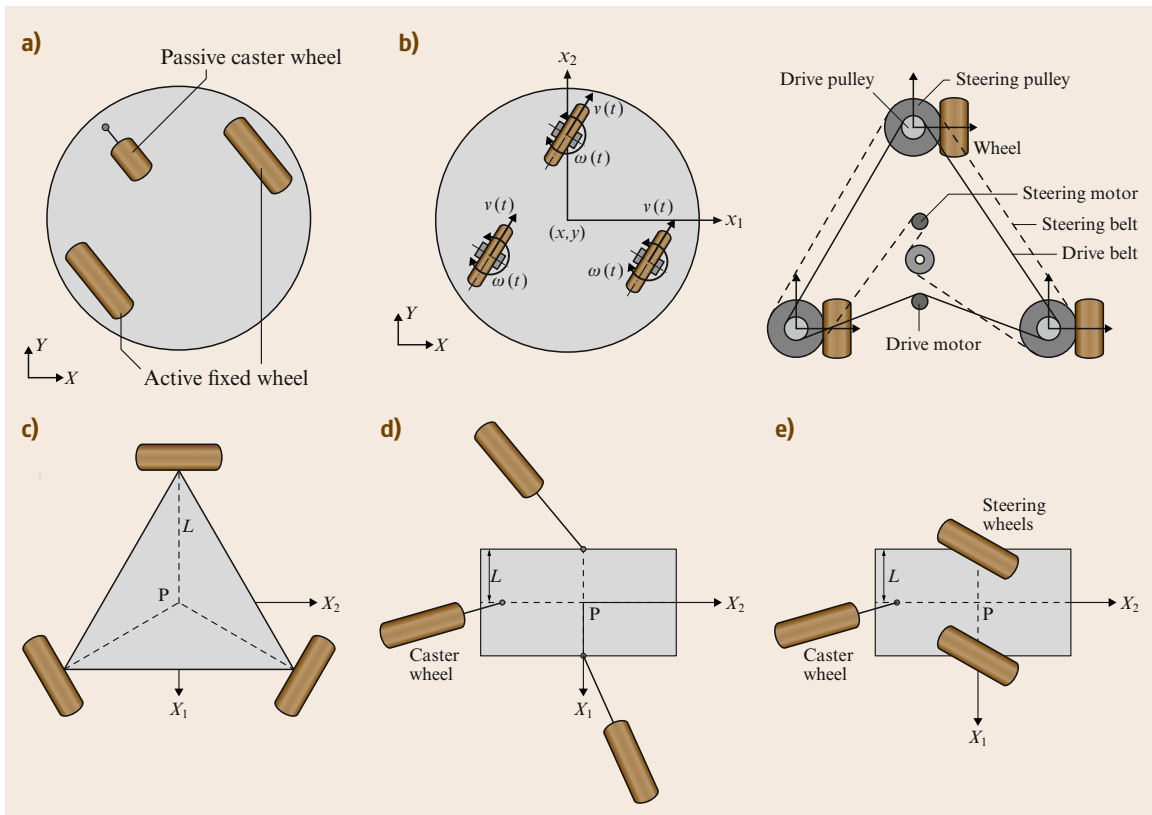


Fig. 17.7 (a) Two-wheel differential drive, (b) synchronous drive, (c) omnimobile robot with Swedish wheels, (d) omnimobile robot with active caster wheels, (e) omnidirectional robot with active steerable wheels

ing and driving motions, less control effort is required for motion control. Other advantages include that odometry information is relatively accurate and driving forces are evenly distributed among all the wheels. The drawbacks of this approach can be summarized as

- complicated mechanical structure
- if backlash or loose coupling is present in the chain transmission, velocity differences between wheels may occur
- in order to achieve omnidirectional movement, the wheel orientations should be aligned to the desired velocity direction before movement, due to the non-holonomic velocity constraints.

Omnimobile Robot with Swedish Wheels

The omnimobile robot with Swedish wheels corresponds to type (3,0) in the nomenclature of Sect. 17.2.6. At least three Swedish wheels are required to build a holonomic omnidirectional robot. A major advantage of using the Swedish wheel is that omnidirectional

mobile robots can be easily constructed. At least three Swedish wheels are required to build a holonomic omnidirectional robot. Since omnidirectional robots can be built without using active steering of wheel modules, the mechanical structures of actuating parts can have simple structures. However, the mechanical design of a wheel becomes slightly complicated. One drawback of the Swedish wheel is that there is a vertical vibration because of discontinuous contacts during motion. In order to solve this problem, a variety of mechanical designs have been proposed; examples can be found in [17.11] and [17.12]. Another drawback is its relatively low durability when compared to conventional tires. An example of a robot using Swedish wheels can be found in [17.13].

Omnimobile Robot with Active Caster Wheels

A holonomic omnidirectional robot can be constructed by using at least two active caster wheels, and the robot

also belongs to type (3,0). The robot can be controlled to generate arbitrary linear and angular velocities regardless of the wheel orientations. Since the robot uses conventional tires, the disadvantages of Swedish wheels, for example, vertical vibrations or durability problems, can be solved. An example can be found in [17.14]. The disadvantages of this robot can be summarized as follows:

- Since the location of the ground contact point (i. e., footprint) changes with respect to the robot chassis, instability can take place when the distance between the wheels is too short.
- If the robot switches its movement to the reverse direction, an abrupt change of wheel orientations may take place. This is called the shopping-cart effect, which may result in instantaneous high steering velocities.
- If a driving motor is directly attached to the wheel, wires to the motor will be wound due to steering motions. In order to avoid this, a gear train should be employed to transmit the input angular velocity from the driving motor, which is attached to the robot chassis. In this case, the mechanical structure becomes quite complicated.
- If a robot is equipped with more than two active caster wheel modules, more than four actuators are used. Since the minimum number of actuator to achieve holonomic omnidirectional motion is three, this is an overactuated system. Therefore, actuators should be accurately controlled in a synchronous way.

Omnidirectional Robot with Steerable Wheels

Centered orientable wheels are also employed to build omnidirectional robots; at least two modules are required. A significant difference between the active caster wheel and the centered orientable wheel is that the wheel orientation should always be aligned with the desired direction of velocity direction, as computed by inverse kinematics. This fact implies that this robot is non-holonomic and omnidirectional: it is a type (1,2) robot. The control problem is addressed in [17.15]. The mechanical drawbacks are similar to those of using active caster wheels (i. e., many actuators and complicated mechanical structures). Since the driving motor is directly attached to the driving axis in many cases, allowable steering angles are limited in order to prevent wiring problems.

There are a lot of design candidates for three-wheel robots, other than the five designs described above.

They can be classified and analyzed according to the scheme presented in Sect. 17.1. The above designs can be extended to four-wheel robots to improve stability. Additional wheels can be passive wheels without adding additional kinematic constraints. Active wheels can also be added and should be controlled by solving the inverse kinematics problem. Four-wheeled robots require suspension to maintain contact with the ground to prevent wheels from floating on irregular surfaces.

17.5.4 Four Robots with Four Wheels

Among the various four-wheel robots, we focus on the car-like structure. The front two wheels should be synchronously steered to keep the same instantaneous center of rotation. Therefore, this solution is kinematically equivalent to a single orientable wheel and the robot can be classified as a type (1,1) robot. A major advantage of a car-like robot is that it is stable during high-speed motion. However, it requires a slightly complicated steering mechanism. If the rear wheels are actuated, a differential gear is required to obtain pure rolling of the rear wheels during the turning motion. If the steering angle of the front wheel cannot reach 90° , the turning radius becomes nonzero. Therefore, parking motion control in a cluttered environment becomes difficult.

17.5.5 Special Applications of Wheeled Robots

Articulated Robots

As explained in Sect. 17.3.5, a robot can be extended to an articulated robot, which is composed of a robot and trailers. A typical example is the luggage-transporting trailer system at airports. By exploiting trailers, a mobile robot obtains various practical advantages. For example, modular and reconfigurable robots can change their configuration according to service tasks. A common design is the car with multiple passive trailers that was presented in Sect. 17.3.5, which is the simplest design of an articulated robot. From the viewpoint of control, some significant issues have been made clear, including a proof of controllability and the development of open- and closed-loop controllers using canonical forms such as the chained form. The design issues for trailer systems are the selection of wheel types and decisions regarding the link parameters. In practical applications, it is advantageous if trailers can move along the path of the towing robot. Passive trailers can follow the path of a towing robot within a small error by using a special

design of passive steering mechanism for trailers; see, for example, [17.16].

On the other hand, active trailers can be used. There are two types of active trailers. A first approach is to actuate wheels of trailers. The connecting joints are passive, and two-wheel differential-drive robots can be used as active trailers. By using this type of active trailers, accurate path-following control can be achieved. The second approach is to actuate connecting joints. The wheels of the trailer are passively driven. By appropriate actuation of the connecting joints, the robot can move without wheel actuation, by snake-like motions. As an alternative design, we can use an active prismatic joint to connect trailers, in order to lift the neighboring trailer. By allowing vertical motion, a trailer system can climb stairs and traverse rough terrain. Examples of active trailers can be found in [17.17].

Hybrid Robots

A fundamental difficulty of using wheels is that they can only be used on flat surfaces. To overcome this problem, wheels are often attached to a special link mechanism. Each wheel is equipped with independent actuators and a linkage mechanism enables the robot to adapt its configuration to irregular ground conditions. A typical design can be found in [17.18] and can be understood as a hybrid robot that is a combination of a legged robot and a wheeled robot. Another hybrid example is a robot equipped with both tracks and wheels. Wheels and tracks have complementary advantages and disadvantages. Wheeled robots are energy efficient, however, tracked robots can traverse rough terrain. Therefore,

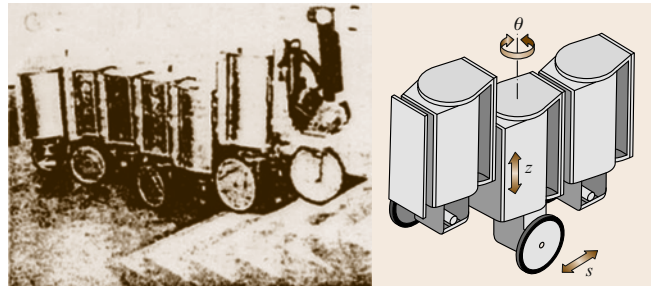


Fig. 17.8 An active trailer system [17.17]

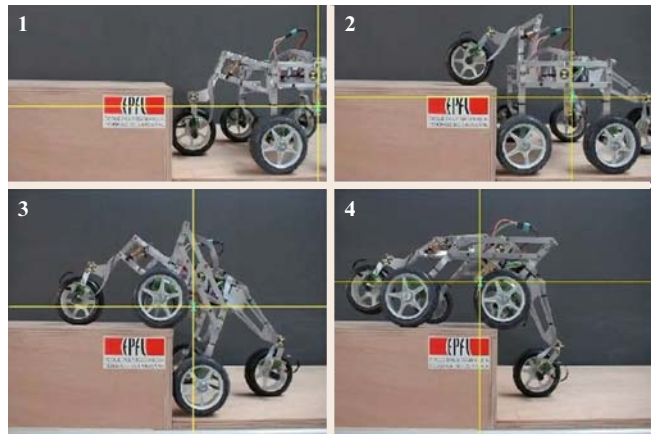


Fig. 17.9 A mobile robot for rough terrain [17.18]

a hybrid robot can selectively choose its driving mechanism according to environmental conditions, although fabrication cost increases.

17.6 Conclusions

The number of possible wheeled mobile robots realizations is almost infinite, depending on the number, type, implementation, geometric characteristics, and motorization of the wheels. This chapter describes several such realizations. Notwithstanding this variety it is possible to classify WMRs into only five generic categories. The (kinematic and dynamic) posture models have exactly the same structure within each of these classes. This fact is crucial for a model-based control design approach, such as that presented in Chap. 34.

The discussion of mobility and the derivation of the models are based on assumptions concerning the contact between the ground and the wheels: it is assumed

that pure rolling and nonslip conditions are satisfied for each wheel. These conditions lead to the kinematic constraints that constitute the basis of the analysis, and particularly of the properties related to the nonholonomy of these models. All model-based control designs therefore also rely on the same assumptions. These assumption are an idealization of the physical reality: these kinematic constraints are not satisfied exactly, and the contact effects are characterized by local slipping effects that are related through phenomenological laws to the contact forces. A question then arises immediately: what is the level of confidence in these models? Using a singular perturbation approach it has been shown that sliding effects can be considered as fast dynamics, with charac-

teristic times that are short compared to the dynamics of the global motion of the WMR and can be neglected, at least for analysis and control design purposes (see, for instance, [17.1]).

References

- 17.1 B. dAndrea-Novet, G. Campion, G. Bastin: Control of wheeled mobile robots not satisfying ideal velocity constraints: a singular perturbation approach, *Int. J. Robust Nonlin. Control* **5**, 243–267 (1995)
- 17.2 H. Asama, M. Sato, L. Bogoni: Development of an omnidirectional mobile robot with 3 DOF decoupling drive mechanism, *Proc. IEEE International conference on robotics and automation* (1995) pp.1925–1930
- 17.3 L. Ferriere, G. Campion, B. Raucourt: ROLLMOBS, a new drive system for omnimobile robots, *Robotica* **19**, 1–9 (2001)
- 17.4 W. Chung: Nonholonomic Manipulators. In: *Springer Tracts Adv Robot*, Vol.13 (Springer, Heidelberg 2004)
- 17.5 J.E. Colgate, M. Peshkin, W. Wannasupphaphasit: Nonholonomic haptic display, *Proc. 1996 IEEE International Conference on Robotics and Automation* (1996) pp. 539–544
- 17.6 G. Campion, G. Bastin, B. dAndrea-Novet: Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, *IEEE Trans. Robot. Automat.* **12**, 47–62 (1996)
- 17.7 R. Nakajima, T. Tsubouchi, S. Yuta, E. Koyanagi: A development of a new mechanism of an autonomous unicycle, *Proc. 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems* (1997) pp. 906–912
- 17.8 G.C. Nandy, X. Yangsheng: Dynamic model of a gyroscopic wheel, *Proc. 1998 IEEE International Conference on Robotics and Automation* (1998) pp.2683–2688
- 17.9 Y. Ha, S. Yuta: Trajectory tracking control for navigation of self-contained mobile inverse pendulum, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems* (1994) pp.1875–1882
- 17.10 Y. Takahashi, T. Takagaki, J. Kishi, Y. Ishii: Back and forward moving scheme of front wheel raising for inverse pendulum control wheel chair robot, *Proc. IEEE International Conference on Robotics and Automation* (2001) pp. 3189–3194
- 17.11 K.-S. Byun, S.-J. Kim, J.-B. Song: Design of continuous alternate wheels for omnidirectional mobile robots, *Proc. IEEE International conference on robotics and automation* (2001) pp. 767–772
- 17.12 M. West, H. Asada: Design and control of ball wheel omnidirectional vehicles, *Proc. IEEE International conference on robotics and automation* (1995) pp.1931–1938
- 17.13 B. Carlisle: An omnidirectional mobile robot. In: *Development in Robotics* (IFS Publ. Ltd., Bedford 1983) pp. 79–87
- 17.14 M. Wada, S. Mori: Holonomic and Omnidirectional Vehicle with Conventional Tires, *Proc. IEEE International conference on robotics and automation* (1996) pp. 3671–3676
- 17.15 D.B. Reister, M.A. Unseren: Position and constraint force control of a vehicle with two or more steerable drive wheels, *IEEE Trans. Robot. Automat.* **9**(6), 723–731 (1993)
- 17.16 Y. Nakamura, H. Ezaki, Y. Tan, W. Chung: Design of steering mechanism and control of nonholonomic trailer systems, *IEEE Trans. Robot. Automat.* **17**(3), 367–374 (2001)
- 17.17 S. Hirose: *Biologically inspired robots: Snake-like Locomotion and Manipulation* (Oxford Univ. Press, Oxford 1993)
- 17.18 R. Siegwart, P. Lamon, T. Estier, M. Lauria, R. Piguat: Innovative design for wheeled locomotion in rough terrain, *J. Robot. Auton. Syst.* **40**, 151–162 (2003)

Micro/Nanorobots

Bradley J. Nelson, Lixin Dong, Fumihito Arai

The field of microrobotics covers the robotic manipulation of objects with dimensions in the millimeter to micron range as well as the design and fabrication of autonomous robotic agents that fall within this size range. Nanorobotics is defined in the same way only for dimensions smaller than a micron. With the ability to position and orient objects with micron- and nanometer-scale dimensions, manipulation at each of these scales is a promising way to enable the assembly of micro- and nanosystems, including micro- and nanorobots.

This chapter overviews the state of the art of both micro- and nanorobotics, outlines scaling effects, actuation, and sensing and fabrication at these scales, and focuses on micro- and nanorobotic manipulation systems and their application in microassembly, biotechnology, and the construction and characterization of micro and nanoelectromechanical systems (MEMS/NEMS). Material science, biotechnology, and micro- and nanoelectronics will also benefit from advances in these areas of robotics.

18.1 Overview of Micro- and Nanorobotics	411
18.2 Scaling	414
18.2.1 The Size of Things.....	414
18.2.2 Predominate Physics at the Micro- and Nanoscales.....	414
18.3 Actuation at the Micro- and Nanoscales .	415
18.3.1 Electrostatics	415
18.3.2 Electromagnetics.....	416
18.3.3 Piezoelectrics.....	416
18.3.4 Other Techniques	417
18.4 Sensing at the Micro- and Nanoscales	417
18.4.1 Optical Microscopy.....	418
18.4.2 Electron Microscopy	418
18.4.3 Scanning Probe Microscopy	418
18.5 Fabrication	419
18.5.1 Microfabrication.....	419
18.5.2 Nanofabrication	421
18.6 Microassembly	422
18.6.1 Automated Microassembly Systems	422
18.6.2 Microassembly System Design	423
18.6.3 Basic Microassembly Techniques ...	426
18.7 Microrobotics	427
18.7.1 Microrobots	428
18.7.2 Bio-microrobotics	429
18.8 Nanorobotics	431
18.8.1 Introduction to Nanomanipulation	432
18.8.2 Nanorobotic Manipulation Systems	434
18.8.3 Nanorobotic Assembly	435
18.8.4 Nanorobotic Devices	442
18.9 Conclusions	443
References	444

18.1 Overview of Micro- and Nanorobotics

Progress in robotics over recent years has dramatically extended our ability to explore, perceive, understand, and manipulate the world on a variety of scales extending from the edges of the solar system, to the bottom of the sea, down to individual atoms (Fig. 18.1). At the lower end of this scale, technology has been

moving toward greater control of the structure of matter, suggesting the feasibility of achieving thorough control of the molecular structure of matter atom by atom, as Richard Feynman first proposed in 1959 in his prophetic article on miniaturization [18.1]:

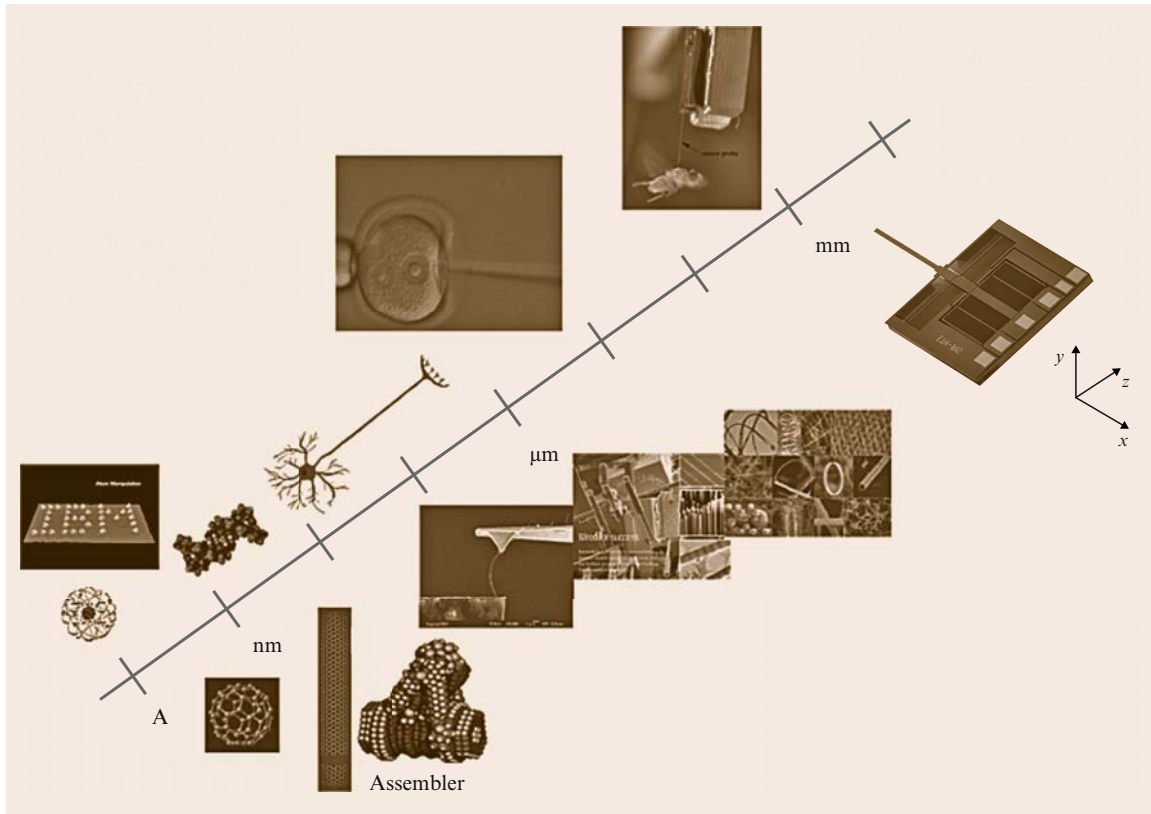


Fig. 18.1 Robotic exploration at micro- and nanometer scales

What I want to talk about is the problem of manipulating and controlling things on a small scale . . . I am not afraid to consider the final question as to whether, ultimately – in the great future – we can arrange the atoms the way we want: the very atoms, all the way down!

He asserted that

At the atomic level, we have new kinds of forces and new kinds of possibilities, new kinds of effects. The problems of manufacture and reproduction of materials will be quite different. The principles of physics, as far as I can see, do not speak against the possibility of maneuvering things atom by atom.

This technology is now labeled *nanotechnology*.

The *great future* of Feynman began to be realized in the 1980s. Some of the capabilities he dreamed of have been demonstrated, while others are being actively pursued. Feynman foresaw the possibility of

employing a microrobotic manipulator (*a master–slave system*) for bottom-up manufacturing (manipulation, assembly, etc.) of minute machines; one such device he described as “swallowing the surgeon”, which he attributed to his friend Albert R. Hibbs. He also imagined we could “build a billion tiny factories, models of each other, which are manufacturing simultaneously, drilling holes, stamping parts, and so on.” Micro- and nanorobotics research has progressed from these seemingly *far-out* concepts of the 1960s and 1970s to reality when microelectromechanical systems (**MEMS**) began to emerge in the late 1980s. These building blocks took the form of surface-micromachined micromotors and microgrippers made of polysilicon fabricated on a silicon chip [18.2]. In the late 1980s and early 1990s, more concrete suggestions on how one could realize **MEMS**-based microrobotic devices using such micromotors as well as potential applications were published [18.3, 4]. Today, a variety of microrobotic devices are enabling new applications in various fields.

In industry, interesting areas for microrobotics include assembly [18.5, 6], characterization, inspection and maintenance [18.7, 8], microoptics (positioning of microoptical chips, microlenses and prisms) [18.9], and microfactories [18.10]. Many of these applications require automated handling and assembly of small parts with accuracy in the submicron range.

Other important fields include biology (manipulation, capturing, sorting and combining cells [18.11]) and medical technology [18.12, 13]. In surgery, the use of steerable catheters and endoscopes is very attractive and the development of increasingly small microrobotic devices is rapidly progressing. Wireless untethered microrobots that will explore and repair our bodies (“swallowing the surgeon”) appear to be simply a matter of time. In fact, endoscopy using wireless capsules (camera pills) are already on the market and allow for endoscopic imaging of the entire gastrointestinal tract [18.14], something currently not possible using standard scopes. Magnetic steering or crawling-type motions serve as promising ways for such devices to locomote in a controlled fashion [18.15]. Doctors could steer pill-mounted cameras and other actuators to areas of interest for visual investigation and biopsies beyond the range of current endoscopes.

Nanorobotics represents the next stage in miniaturization for maneuvering nanoscale objects. Nanorobotics is the study of robotics at the nanometer scale, and includes robots that are nanoscale in size, i. e., nanorobots, and large robots capable of manipulating objects that have nanometer dimensions with nanometer resolution, i. e., nanorobotic manipulators. The field of nanorobotics brings together several disciplines, including nanofabrication processes used for producing nanoscale robots, nanoactuators, nanosensors, and physical modeling at nanoscales. Nanorobotic manipulation technologies, including the assembly of nanometer-sized parts, the manipulation of biological cells or molecules, and the types of robots used to perform these types of tasks also form a component of nanorobotics.

As the 21st century unfolds, the impact of nanotechnology on the health, wealth, and security of humankind is expected to be at least as significant as the combined influences in the 20th century of antibiotics, the integrated circuit, and human-made polymers. For example, N. Lane stated in 1998, “If I were asked for an area of science and engineering that will most likely produce the breakthroughs of tomorrow, I would point to nanoscale science and engineering.” [18.16] The great scientific and technological opportunities nanotechnol-

ogy presents have stimulated extensive exploration of the nanoworld and initiated an exciting worldwide competition, which has been accelerated by the publication of the *National Nanotechnology Initiative* by the US government in 2000 [18.17]. Nanorobotics will play a significant role as an enabling nanotechnology and could ultimately be a core part of nanotechnology if Drexler’s machine-phase nanosystems based on self-replicative molecular assemblers via mechanosynthesis can be realized [18.18].

By the early 1980s, scanning tunneling microscopes (STMs) [18.19] radically changed the way in which we interacted with and even regarded single atoms and molecules. The very nature of proximal probe methods encourages the exploration of the nanoworld beyond conventional microscopic imaging. Scanning probes now allow us to perform *engineering* operations on single molecules, atoms, and bonds, thereby providing a tool that operates at the ultimate limits of fabrication. They have also enabled exploration of molecular properties on an individual nonstatistical basis.

STMs and other nanomanipulators are nonmolecular machines that use bottom-up strategies. Although performing only one molecular reaction at a time is obviously impractical for making large amounts of a product, it is a promising way to provide the next generation of nanomanipulators. Most important, it is possible to realize the directed assembly of molecules or supermolecules to build larger nanostructures through nanomanipulation. The products produced by nanomanipulation could be the first step of a bottom-up strategy in which these assembled products are used to self-assemble into nanomachines.

One of the most important applications of nanorobotic manipulation will be nanorobotic assembly. However, it appears that, until assemblers capable of replication can be built, the combination of chemical synthesis and self-assembly are necessary when starting from atoms; groups of molecules can self-assemble quickly due to their thermal motion, enabling them to *explore* their environments and find (and bind to) complementary molecules. Given their key role in natural molecular machines, proteins are obvious candidates for early work in self-assembling artificial molecular systems. *Degrado* [18.20] demonstrated the feasibility of designing protein chains that predictably fold into solid molecular objects. Progress is also being made in artificial enzymes and other relatively small molecules that perform functions like those of natural proteins. Several bottom-up strategies using self-assembly appear feasible [18.21]. Chemical synthesis, self-assembly, and

supramolecular chemistry make it possible to provide building blocks at relatively large sizes beginning from the nanometer scale. Nanorobotic manipulation serves as the base for a hybrid approach to construct nanodevices by structuring these materials to obtain building blocks and assembling them into more complex systems.

Despite the claims of many *futurists*, the form nanorobots of the future will take and what tasks they will actually perform remain unclear. However, it is clear that nanotechnology is progressing towards the construction of intelligent sensors, actuators, and systems that are smaller than 100 nm. These nanoelectromechanical systems (NEMS) will serve as both the tools to be used for fabricating future nanorobots as well as the components from which these nanorobots may be developed. Shrinking device size to these dimensions presents many fascinating opportunities such as manipulating nano-objects with nanotools, measuring mass in

femtogram ranges, sensing forces at piconewton scales, and inducing gigahertz motion, among other new possibilities waiting to be discovered. These capabilities will, of course, drive the tasks that future nanorobots constructed by and with NEMS will perform. NEMS and the components of nanorobots will be the products of nanorobotic manipulation. Large nanorobotic manipulators will be able to shrink in size due to this development, thus enabling nanosized robotic manipulators and other forms of nanorobots. All of these form the scope of the area of nanorobotics.

This chapter focuses on micro- and nanorobotics including actuation, manipulation, and assembly at the micro- and nanoscale. The main goal of these fields of robotics is to provide an effective technology for the experimental exploration of the micro- and nanoworld, and to push the boundaries of this exploration from a robotics research perspective.

18.2 Scaling

18.2.1 The Size of Things

Things we can potentially observe range from 10^{-35} m (the Planck length) to 10^{26} m (the radius of the observable universe). A nanometer, 10^{-9} m, is about ten times the size of the smallest atoms, such as hydrogen and carbon, while a micron is barely larger than the average wavelength of visible light, thus invisible to the human eye. A millimeter, the size of a pinhead, is roughly the smallest part typically fabricated using traditional machining techniques. The range of scales from millimeters to nanometers is one million (Fig. 18.1), which is also about the range of scales in present-day mechanical technology from the largest skyscrapers to the smallest conventional mechanical machine parts. The vast opportunity to make new machines spanning almost six orders of magnitude from 1 mm to 1 nm, is one take on Richard Feynman's famous statement, "there is plenty of room at the bottom" [18.1]. If L is taken as a typical length, 0.1 nm for an atom, perhaps 2 m for a human, this scale range in L would be 2×10^{10} . If the same scale range were to apply to an area, $0.1 \text{ nm} \times 0.1 \text{ nm}$ versus $2 \text{ m} \times 2 \text{ m}$, the scale range for area L^2 is 4×10^{20} . Since a volume L^3 is enclosed by sides L , we can see that the number of atoms of size 0.1 nm in a $(2 \text{ m})^3$ volume is about 8×10^{30} , recalling that Avogadro's number $N_A = 6.022 \times 10^{23}$ is the number of atoms in a gram-mole, supposing that the atoms were ^{12}C , mo-

lar responding to a density $1.99 \times 10^4 \text{ kg/m}^3$. A primary working tool of the nanotechnologist is facility in scaling the magnitudes of various properties of interest, as the length scale L shrinks, e.g., from 1 mm to 1 nm.

Clearly, the number of atoms in a device scales as L^3 . If a transistor on the micron scale contains 10^{12} atoms, then on the nanometer scale, $L'/L = 10^{-3}$ it will contain 1000 atoms, likely too few to preserve its function.

Normally, we will think of scaling as an isotropic scale reduction in three dimensions. However, scaling can be thought of usefully when applied only to one or two dimensions, scaling a cube to a two-dimensional (2-D) sheet of thickness a or to a one-dimensional (1-D) tube or nanowire of cross-sectional area a^2 . The term *zero-dimensional* (0-D) is used to describe an object small in all three dimensions, having volume a^3 . In electronics, a zero-dimensional object (a nanometer-sized cube a^3 of semiconductor) is called a quantum dot (QD) or *artificial atom* because its electronic states are few and sharply separated in energy, and thus resemble the electronic states of an atom.

18.2.2 Predominate Physics at the Micro- and Nanoscales

The predominate physics at the micro- and nanoscale can be dramatically different from at the macroscale. Surface and intermolecular forces, such as adhesion forces

originating from surface tension forces, van der Waals forces, and electrostatic forces, become more significant than volumetric forces such as gravitational forces for objects with sizes well below $1000\text{ }\mu\text{m}$ [18.22]. Although the laws of classical Newtonian physics may well suffice to describe changes in behavior down to 10 nm ($100\text{ }\text{\AA}$), the range of scaling is tremendous. Therefore, the changes in magnitudes of many important physical properties, such as resonant frequencies, are so great, that completely different applications may appear.

The more challenging question for the nanotechnologist is to understand and hopefully to exploit those changes in physical behavior that occur at the end of the classical scaling range. The *end of the scaling* is the size scale of atoms and molecules, where nanophysics [18.23] is the proven conceptual replace-

ment of the laws of classical physics. Modern physics, which includes quantum mechanics as a description of matter on a nanometer scale, is a very well-developed and proven subject whose application to real situations is limited only by modeling and computational competence.

In the modern era, simulations and approximate solutions increasingly facilitate the application of nanophysics to almost any problem of interest. Many central problems are already (adequately, or more than adequately) solved in the extensive literatures of theoretical chemistry, biophysics, condensed matter physics, and semiconductor device physics. The practical problem is to find the relevant work, and, frequently, to convert the notation and units systems to apply the results to the problem at hand.

18.3 Actuation at the Micro- and Nanoscales

The positioning of nanorobots and nanorobotic manipulators depends largely on nanoactuators. While nanosized actuators for nanorobots are still under exploration and relatively far from implementation, MEMS-based efforts are focused on shrinking their sizes [18.30]. Nanometer resolution motion has been extensively investigated and can be generated using various actuation principles. Electrostatics, electromagnetics, and piezoelectrics are the most common ways to realize actuation at nanoscales. For nanorobotic manipulation, besides nanoresolution and compact sizes, actuators generating large strokes and high forces are best suited for such applications. The speed criteria are of less importance as long as the actuation speed is in the range of a couple of hertz and above. Table 18.1 provides a small selection of early works on actua-

tors [18.24–29] suitable for micro- and nanorobotic applications (partially adapted from [18.30]).

Several extensive reviews on various actuation principles have been published [18.4, 31–34]. During the design of an actuator, the trade-offs among range of motion, force, speed (actuation frequency), power consumption, control accuracy, system reliability, robustness, load capacity, etc. must be taken into consideration. This section reviews basic actuation technologies and potential applications at nanometer scales.

18.3.1 Electrostatics

Electrostatic charge arises from a build up or deficit of free electrons in a material, which can exert an attractive force on oppositely charged objects, or a repulsive

Table 18.1 Actuation with MEMS

Actuation principle	Type of motion	Volume (mm^3)	Speed (s^{-1})	Force (N)	Stroke (m)	Resolution (m)	Power density (W/m^3)	Ref.
Electrostatic	Linear	400	5000	1×10^{-7}	6×10^{-6}	NA	200	[18.24]
Magnetic	Linear	$0.4 \times 0.4 \times 0.5$	1000	2.6×10^{-6}	1×10^{-4}	NA	3000	[18.25]
Piezoelectric	Linear	$25.4 \times 12.7 \times 1.6$	4000	350	1×10^{-3}	7×10^{-8}	NA	[18.26]
Actuation principle	Type of motion	Volume (mm^3)	Speed (rad/s)	Torque (Nm)	Stroke (rad)	Resolution (rad)	Power density (W/m^3)	Ref.
Electrostatic	Rotational	$\pi/4 \times 0.5^2 \times 3$	40	2×10^{-7}	2π	NA	900	[18.27]
Magnetic	Rotational	$2 \times 3.7 \times 0.5$	150	1×10^{-6}	2π	$5/36\pi$	3000	[18.28]
Piezoelectric	Rotational	$\pi/4 \times 1.5^2 \times 0.5$	30	2×10^{-11}	0.7	NA	NA	[18.29]

force on similarly charged objects. Since electrostatic fields arise and disappear rapidly, such devices will likewise demonstrate very fast operation speeds and be little affected by ambient temperatures.

Previous investigations have produced many examples of miniature devices using electrostatic force for actuation including silicon micromotors [18.35,36], microvalves [18.37], and microtweezers [18.38]. This type of actuation is important for achieving nanoscale actuation.

Electrostatic fields can exert great forces, but generally across very short distances. When the electric field must act over larger distances, a higher voltage will be required to maintain a given force. The extremely low current consumption associated with electrostatic devices makes for highly efficient actuation.

18.3.2 Electromagnetics

Electromagnetism arises from electric current moving through a conducting material. Attractive or repulsive forces are generated adjacent to the conductor and proportional to the current flow. Structures can be built which gather and focus electromagnetic forces, and harness these forces to create motion.

Electromagnetic fields arise and disappear rapidly, thus permitting devices with very fast operation speeds. Since electromagnetic fields can exist over a wide range of temperatures, performance is primarily limited by the properties of the materials used in constructing the actuator.

One example of a microfabricated electromagnetic actuator is a microvalve which uses a small electromagnetic coil wrapped around a silicon micromachined valve structure [18.39]. The downward scalability of electromagnetic actuators into the micro- and nano-

realm may be limited by the difficulty of fabricating small electromagnetic coils. Furthermore, most electromagnetic devices require perpendicularity between the current conductor and the moving element, presenting a difficulty for planar fabrication techniques commonly used to make silicon devices.

An important advantage of electromagnetic devices is their high efficiency in converting electrical energy into mechanical work. This translates into less current consumption from the power source.

18.3.3 Piezoelectrics

Piezoelectric motion arises from the dimensional changes generated in certain crystalline materials when subjected to an electric field or to an electric charge. Structures can be built which gather and focus the force of the dimensional changes, and harness them to create motion. Typical piezoelectric materials include quartz (SiO₂), lead zirconate titanate (PZT), lithium niobate, and polymers such as polyvinylidene fluoride (PVDF).

Piezoelectric materials respond very quickly to changes in voltages and with great repeatability. They can be used to generate precise motions with repeatable oscillations, as in quartz timing crystals used in many electronic devices. Piezoelectric materials can also act as sensors, converting tension or compression strains to voltages.

On the microscale, piezoelectric materials have been used in linear inchworm drive devices and micropumps [18.40]. STMs and most nanomanipulators use piezoelectric actuators.

Piezoelectric materials operate with high force and speed, and return to a neutral position when unpowered. They exhibit very small strokes (under 1%). Alternating electric currents produce oscillations in the piezoelec-

Table 18.2 Comparison of nanoactuators

Method	Efficiency	Speed	Power density
Electrostatic	Very high	Fast	Low
Electromagnetic	High	Fast	High
Piezoelectric	Very high	Fast	High
Thermomechanical	Very high	Medium	Medium
Phase change	Very high	Medium	High
Shape memory	Low	Medium	Very high
Magnetostrictive	Medium	Fast	Very high
Electrorheological	Medium	Medium	Medium
Electrohydrodynamic	Medium	Medium	Low
Diamagnetism	High	Fast	High

tric material, and operation at the sample's fundamental resonant frequency produces the largest elongation and highest power efficiency [18.41]. Piezoelectric actuators working in the *stick-slip* mode can provide millimeter to centimeter strokes. Most commercially available nanomanipulators adopt this type of actuators, such as Picomotors™ from New Focus and Nanomotors™ from Klock.

18.3.4 Other Techniques

Other techniques include thermomechanical, phase change, shape memory, magnetostrictive, electrorheological, electrohydrodynamic, diamagnetism, magnetohydrodynamic, shape changing, polymers, and biological methods (living tissues, muscle cells, etc.). Table 18.2 compares these techniques.

18.4 Sensing at the Micro- and Nanoscales

A brief overview is given of common imaging tools used in the research area of micro- and nanorobotics including optical, electron, and scanning probe microscopy. The application and integration of these tools for micro- and nanorobotics are discussed in separate sections.

In selecting a proper tool for imaging/sensing at these scales, the following factors should be first considered:

1. Specimen: size, conductivity, and environment compatibility are among the most important aspects to consider. For example, in vivo bioapplications generally require air or liquid, so lower-resolution, light microscopy should be the first choice. If higher resolution is needed, atomic force microscopy or scanning near-field optical microscopy can be used.
2. Resolution: the ability to see fine details of a specimen. Once you can resolve fine details then you can magnify them. Every microscope has a finite resolution; if you magnify objects beyond the resolution the result will be empty magnification. Roughly speaking, a light microscope cannot provide a resolution better than 200 nm. The best commercially available scanning electron microscopes (SEM) have approximately 1 nm resolution, while transmission electron microscopes (TEM) can achieve approximately 0.2 nm, and a scanning tunneling microscope (STM) working in ultrahigh vacuum under very low temperature can resolve atomic-level structures. A comparison of optical microscopy and electron microscopy is given in Table 18.3.
3. Depth of field: the range of depth that a specimen is in acceptable focus. A microscope that has a small depth of field will have to be continuously focused up and down to view a thick specimen.
4. Contrast: the ratio between dark and light. Typically, most microscopes use absorption contrast, i.e., the specimen is subjected to stains in order to be seen. This is called bright-field microscopy. There are other types of microscope that use more exotic means to generate contrast, such as phase contrast, dark field, and differential interference contrast.
5. Brightness: the amount of light. The higher a microscope magnifies the more light will be required. The illumination source should also be at a wavelength (color) that will facilitate interaction with the specimen. All microscopes fall into either of two cat-

Table 18.3 Comparison of optical microscopes and electron microscopes

Feature	OM	TEM	SEM
General use	Surface morphology and sections (1–40 μm)	Section (40–150 nm) or small particles on thin membranes	Surface morphology
Source of illumination	Visible light	High-speed electrons	High-speed electrons
Best resolution	ca. 200 nm	ca. 0.2 nm	ca. 3–6 nm
Magnification range	10–1000 \times	500–500 000 \times	20–150 000 \times
Depth of field	0.002–0.05 nm (N.A. 1.5)	0.004–0.006 mm (N.A. 10^{-3})	0.003–1 mm
Lens type	Glass	Electromagnetic	Electromagnetic
Image ray-formation spot	On eye by lenses	On phosphorescent plate by lenses	On cathode tube by scanning device

egories based on how the specimen is illuminated. In the typical compound microscope the light passes through the specimen and is collected by the image forming optics. This is called diascope illumination. Dissecting (stereo) microscopes generally use episcope illumination for use with opaque specimen. The light is reflected onto the specimen and then into the objective lens.

18.4.1 Optical Microscopy

Since their invention in the late 1500s, light microscopes have enhanced our knowledge in basic biology, biomedical research, medical diagnostics, and materials science. Light microscopes can magnify objects up to 1000 times, revealing microscopic details. Light-microscopy technology has evolved far beyond the first microscopes of Robert Hooke and Antoni van Leeuwenhoek. Special techniques and optics have been developed to reveal the structures and biochemistry of living cells. Most optical microscopes in current use are known as compound microscopes, where a magnified image of an object is produced by the objective lens, and this image is magnified by a second lens system (the ocular or eyepiece) for viewing. Microscopes have even entered the digital age, using charge-coupled devices (CCDs) and digital cameras to capture images.

The development of modern microscopy has enabled a large family of optical microscopes. For special purposes, other types of optical microscopes can be selected. These include phase-contrast microscopy, fluorescence microscopy, confocal scanning optical microscopy, and deconvolution microscopy image reconstruction.

18.4.2 Electron Microscopy

Scanning Electron Microscope (SEM)

Since the commercial availability of the SEM in 1966, it has been a valuable resource for viewing samples at a much higher resolution and depth of field than the typical optical microscope. Conventional SEMs can resolve down to the nanometer scale (≈ 1 nm) whereas an optical microscope can only resolve down to approximately 200 nm [18.42]. Unlike conventional optical microscopes, SEMs have a high depth of field which gives imaged samples a three-dimensional appearance. Early SEMs were limited to viewing conductive samples. However, many of today's SEMs can image nonconductive samples in addition to conductive samples using variable-pressure chambers.

Transmission Electron Microscope (TEM)

The TEM can resolve to an atomic scale down to about 1 Å (i. e., 0.1 nm). The TEM mode of operation is similar to that of the SEM in that both microscopes contain an electron gun source of illumination. However, the TEM detects the electrons that pass through a given sample. As a result, the electron gun of the TEM operates at higher energy levels between 50–1000 kV, while the SEM's electron gun operates at around 1–30 kV. In order for proper imaging to take place, the sample must be very thin so that electrons from the beam can pass through the specimen. Electrons that do not pass through the sample cannot be detected. Unlike the SEM, the TEM produces images that are two dimensional in appearance.

18.4.3 Scanning Probe Microscopy

Scanning Tunneling Microscope (STM)

Similar to the TEM, the STM [18.19] can also resolve specimens down to the atomic scale. The scanning probe of the STM is comprised of a noble metal sharpened to an atomic-sized tip, which is mounted on a piezoelectrically driven (x, y, z) linear stage. The STM makes use of the quantum-mechanical effect known as tunneling. Electron tunneling occurs when electrons, driven by a small potential difference, flow across the gap between the probe tip and sample. This event takes place at Ångström-scale distances between the probe tip and the sample [18.43]. The tunneling current, which is typically on the order of a few nanoamperes, is directly related to the tip-sample separation distance. Thus, the tunneling current can be measured and is kept at a constant value by controlling the tip-sample gap distance (z) with a feedback control system. The probe tip is then scanned (x, y) along the entire surface of the sample. Since the control system maintains a constant tunneling current, and thus maintains a constant tip-sample distance (z), the result of a scan yields a $z(x, y)$ terrain map of the sample with enough resolution to detect atomic-scale features. The STM can achieve faster imaging by operating in so-called constant-height mode in which the probe tip is scanned in a plane parallel to the average surface portion. The tip-sample distance (z) can then be inferred directly from the measured tunneling current [18.43].

Atomic Force Microscope (AFM)

The AFM [18.44] is considered to be a spin-off of the STM. One shortcoming of the STM is that it requires conductive probe tips and samples to work properly. The AFM was developed in order to view nonconduc-

tive samples, giving it a wider applicability than the **STM**. In addition to imaging nonconductive samples, the AFM can also image samples immersed in liquid, which is useful for biological applications [18.45]. Although the **STM** and AFM are similar in that they both scan the surface of a sample with an atomically sharp probe, they operate under slightly different principles. The AFM is based on interatomic forces as opposed to the electron tunneling used in the **STM**. The AFM probe tip is mounted on the end of a microscale cantilever beam. At very short separations, the forces between atoms in the probe tip and atoms in the sample cause the cantilever to deflect. This deflection is usually measured by striking the back of the cantilever with a laser. The reflection of the laser beam hits a photodetector, which can be used to recover the deflection of the cantilever. The force can then be cal-

culated by using Hooke's law, which simply relates the applied force to the stiffness and deflection of a material. The forces that are measured can be on the scale of piconewtons [18.43]. The AFM has three main modes of operation known as contact mode, noncontact mode, and tapping mode.

Unlike the **SEM** and **TEM**, both the **STM** and AFM do not require a vacuum environment in order to function. However, a high vacuum is advantageous in order to keep the samples from becoming contaminated from the surrounding environment as well as controlling humidity. In addition, atomic resolution in air is hardly possible with an AFM due to humidity. As a result of humidity, a water film is formed and creates problems because of capillary forces. This can be resolved by operating in vacuum environment or completely in a liquid solution [18.43].

18.5 Fabrication

The design of micro- and nanorobotic devices is inextricably linked to available fabrication techniques. However, though the development of microfabrication processes has become somewhat stable over the past decade, nanofabrication processes are still being actively pursued, and the design constraints generated by these processes are relatively unexplored. This section will briefly highlight the processes used in conventional microfabrication including lithography, thin-film deposition, chemical etching, and electrodeposition, and describe some emerging techniques for nanofabrication.

Most micro- and nanofabrication techniques have their roots in the standard fabrication methods developed for the semiconductor industry [18.46–48]. Therefore, a clear understanding of these techniques is necessary for anyone embarking on a research and development path in the micro/nano area.

18.5.1 Microfabrication

In this section, we will discuss the major microfabrication techniques used most frequently in the manufacturing of microstructures.

Photolithography

Lithography is the technique used to transfer a computer-generated pattern onto a substrate (silicon, glass, GaAs, etc.). This pattern is subsequently used to etch an under-

lying thin film (oxide, nitride, etc.) for various purposes (doping, etching, etc.). Although photolithography, i. e., lithography using an ultraviolet (UV) light source, is by far the most widely used lithography technique in microelectronic fabrication, electron-beam (e-beam) and X-ray lithography are two alternatives that have attracted considerable attention in the **MEMS** and nanofabrication areas. We will discuss photolithography in this section and postpone the discussion of e-beam and X-ray techniques to the subsequent sections dealing with nanofabrication.

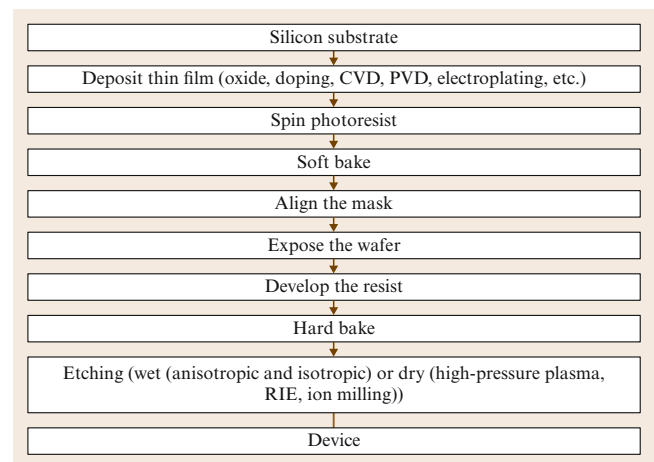


Fig. 18.2 Typical microfabrication process flow

The starting point following the creation of the computer layout for a specific fabrication sequence is the generation of a photomask. This involves a sequence of photographic processes (using optical or e-beam pattern generators) that results in a glass plate having the desired pattern in the form of a thin (≈ 100 nm) chromium layer. Following the generation of photomask, the lithography and etching process can proceed as shown in Fig. 18.2. After depositing the desired material on the substrate, the photolithography process starts with spin-coating the substrate with a photoresist. This is a polymeric photosensitive material that can be spun onto the wafer in liquid form; usually an adhesion promoter such as hexamethyldisilazane (HMDS) is used prior to the application of the resist. The spinning speed and photoresist viscosity will determine the final resist thickness, which is typically $0.5\text{--}2.5\text{ }\mu\text{m}$. Two different kinds of photoresist are available: positive and negative. With a positive resist, the UV-exposed areas will be dissolved in the subsequent development stage, whereas with a negative photoresist, the exposed areas will remain intact after the development. After spinning the photoresist on the wafer, the substrate is soft-baked ($5\text{--}30$ min at $60^{\circ}\text{--}100^{\circ}\text{C}$) in order to remove the solvents from the resist and improve the adhesion. Subsequently, the mask is aligned to the wafer and the photoresist is exposed to a UV source.

After exposure, the photoresist is developed in a process similar to the development of photographic film. The resist is subsequently hard-baked ($20\text{--}30$ min at $120^{\circ}\text{--}180^{\circ}\text{C}$) in order to further improve the adhesion. The hard-bake step concludes the photolithography sequence by creating the desired pattern on the wafer. Next, the underlying thin film is etched, and the photoresist is stripped in acetone or other organic solvent. Figure 18.3 shows a schematic of the photolithography steps with a positive photoresist.

Thin-Film Deposition and Doping

Thin-film deposition and doping are used extensively in micro- and nanofabrication technologies. Most of the fabricated structures contain materials other than that of the substrate, which are obtained by various deposition techniques, or by modification of the substrate. These techniques include oxidation, doping, chemical vapor deposition (CVD), physical vapor deposition (PVD), and electroplating.

Etching and Substrate Removal

For micro- and nanofabrication, in addition to thin film etching, often the substrate (silicon, glass, GaAs, etc.)

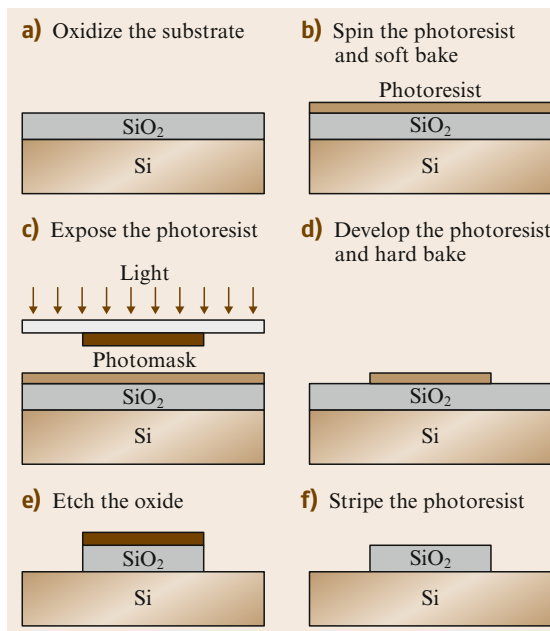


Fig. 18.3a–f Schematic drawing of the photolithographic steps with a positive photoresist (PR)

needs to be removed in order to create various mechanical structures (beams, plates, etc.). Two important figures of merit for any etching process are selectivity and directionality.

Selectivity is the degree to which the etchant can differentiate between the masking layer and the layer to be etched. Directionality has to do with the etch profile under the mask. In an isotropic etch, the etchant attacks the material in all directions at the same rate, creating a semicircular profile under the mask (Fig. 18.4a). In an anisotropic etch, the dissolution rate depends on specific directions, and one can obtain straight sidewalls or other noncircular profiles (Fig. 18.4b). One can also divide the various etching techniques into wet and dry categories. Due to the lateral undercut, the minimum feature size achievable with wet etchants is limited to $> 3\text{ }\mu\text{m}$. Photoresist and silicon nitride are the two most common masking materials for the wet oxide etch. Anisotropic and isotropic wet etching of crystalline (silicon and gallium arsenide) and noncrystalline (glass) substrates are important topics in micro- and nanofabrication [18.49–53].

The anisotropic behavior of these etchants with respect to the (111) planes have been used extensively to create beams, membranes, and other mechanical and structural components. Figure 18.5 shows the typical

cross sections of (100) silicon wafers etched with an anisotropic wet etchant. As can be seen, the (111) slow planes are exposed and creating 54.7° sloped sidewalls. Depending on the dimensions of the mask opening, a V-groove or a trapezoidal trench is formed in the (100) wafer. A large enough opening will allow the silicon to be etched all the way through the wafer, thus creating a thin dielectric membrane of the other side. It should be mentioned that exposed convex corners have a higher etch rate than the concave ones, resulting in an undercut that can be used to create dielectric (e.g., nitride) cantilever beams.

Dry etching techniques are largely plasma based. They have several advantages compared with wet etching. These include smaller undercut (allowing smaller lines to be patterned) and higher anisotropy (allowing high-aspect-ratio vertical structures). However, the selectivity of dry etching techniques is lower than the wet etchants, and one must take into account the finite etch rate of the masking materials. The three basic dry etching techniques, namely high-pressure plasma etching, reactive-ion etching (RIE), and ion milling, utilize different mechanisms to obtain directionality.

18.5.2 Nanofabrication

The design and fabrication of **NEMS** is an emerging area being pursued by an increasing number of researchers. Two approaches to nanofabrication, top-down and bottom-up, have been identified by the nanotechnology research community and are being independently investigated by various researchers. Top-down approaches are based on microfabrication and include technologies such as nanolithography, nanoimprinting, and chemical etching. Presently, these are 2-D fabrication processes with relatively low resolution. Bottom-up strategies are assembly-based techniques. Currently these strategies include techniques such as self-assembly, dip-pen lithography, and directed self-assembly. These techniques can generate regular nanopatterns at large scales.

In this section, we will discuss three major nanofabrication techniques. These include:

1. e-beam and *nanoimprint* fabrication,
2. *epitaxy* and *strain engineering*, and
3. dip-pen nanolithography.

E-beam Lithography and Nanoimprint Fabrication

In previous sections, we discussed several important lithography techniques commonly used in **MEMS** and

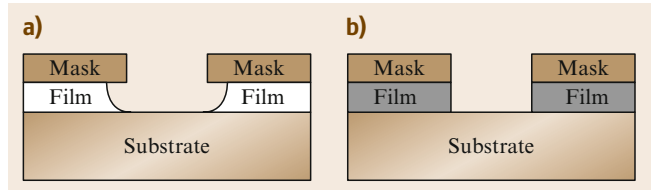


Fig. 18.4a,b Profile for isotropic (a) and anisotropic (b) etch through a photoresist mask

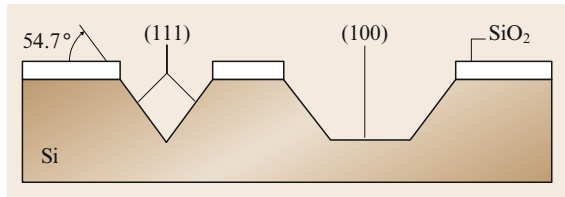


Fig. 18.5 Anisotropic etch profiles for (100) silicon wafers

microfabrication. These include various forms of UV (regular, deep, and extreme) and X-ray lithography. However, due to the lack of resolution (in the case of the UV), or the difficulty in manufacturing mask and radiation sources (X-ray), these techniques are not suitable for nanometer-scale fabrication. E-beam lithography is an attractive alternative technique for fabricating nanostructures [18.53]. It uses an electron beam to expose an electron-sensitive resist such as polymethyl methacrylate (**PMMA**) dissolved in trichlorobenzene (positive) or polychloromethylstyrene (negative).

The e-beam gun is usually part of an **SEM**, although a **TEM** can also be used. Although electron wavelengths on the order of 1 \AA can easily be achieved, electron scattering in the resist limits the attainable resolutions to $> 10 \text{ nm}$. The beam control and pattern generation are achieved through a computer interface.

E-beam lithography is serial and hence has a low throughput. Although this is not a major concern in fabricating devices used in studying fundamental microphysics, it severely limits large-scale nanofabrication. E-beam lithography, in conjunction with such processes as lift-off, etching, and electrodeposition, can be used to fabricate various nanostructures.

An interesting new technique that circumvents the serial and low-throughput limitations of the e-beam lithography for fabricating nanostructures is nanoimprint technology [18.54]. This technique uses an e-beam-fabricated hard material master (or mold) to stamp and deform a polymeric resist. This is usually followed by a reactive-ion etching step to transfer the stamped pattern to the substrate. This technique is eco-

onomically superior, since a single stamp can be used repeatedly to fabricate a large number of nanostructures.

Epitaxy and Strain Engineering

Atomic-precision deposition techniques such as molecular-beam epitaxy (MBE) and metallo-organic chemical vapor deposition (MOCVD) have proven to be effective tools in fabricating a variety of quantum confinement structures and devices (quantum well lasers, photodetectors, resonant tunneling diodes, etc.) [18.55–57].

18.6 Microassembly

Assembly is often required in macroscale product manufacturing in order to reduce the complexity and cost of the manufacturing process. Assembly makes it possible to build complex products from relatively simple parts and to integrate incompatible manufacturing processes. It also makes maintenance and replacement possible. The extension of assembly techniques into the microscale is driven by the development of modern design and manufacturing technologies in the pursuit of miniaturization and function integration, especially by the development of integrated circuit (IC) [18.59] and MEMS fabrication techniques. With the extension of manufacturing technology into the microscale and even nanoscale domain, the term microassembly has been created to refer specifically to assembly operations performed at the micro/mesoscale [18.60]. To give a formal definition, microassembly is the assembly of objects with microscale and/or mesoscale features under microscale tolerances [18.61].

18.6.1 Automated Microassembly Systems

Microassembly plays the role of an enabling technology in various processes of MEMS fabrication, including device fabrication, packaging, and interconnection. MEMS device fabrication is fundamentally different from the highly modular IC fabrication in that it often requires the machining of complex-shaped 3-D mechanical structures [18.51]. However, almost all current MEMS fabrication techniques are subject to constraints in limited allowable materials, limited capabilities in true 3-D fabrication, and the requirement of fabrication process compatibility. Microassembly provides a possible solution to these constraints. For example, incompatible fabrication processes can be integrated

Dip-Pen Nanolithography

In dip-pen nanolithography (DPN), the tip of an AFM operated in air is *inked* with a chemical of interest and brought into contact with a surface. The ink molecules flow from the tip onto the surface as with a fountain pen. Line widths down to 12 nm with spatial resolution of 5 nm have been demonstrated with this technique [18.58]. Species patterned with DPN include conducting polymers, gold, dendrimers, deoxyribonucleic acid (DNA), organic dyes, antibodies, and alkanethiols.

through assembly. This makes it possible to use *nontraditional* fabrication techniques that are not necessarily based on semiconductor materials, such as laser cutting, microwire electrical discharge machining (EDM), and micromilling [18.62]. Complex 3-D structures can also be developed using parts with relatively simple geometry [18.63, 64]. Microassembly is also crucial to MEMS packaging and interconnection [18.65, 66].

From the perspective of robotic systems and automation, MEMS device fabrication and packaging share many common assembly requirements. A fundamental commonality of both processes is the requirement of the ability to manipulate micro/mesoscale objects so that precise (i. e., microscale tolerance) spatial relations can be established (e.g., die alignment, part insertion) and certain physical/chemical processes (e.g., die bonding, surface coating) can be performed. Another common requirement is to control the interaction force involved. MEMS devices often have fragile structures such as thin beams or membranes. This requires controlling the interactive force in manipulation operations. Typically, the force magnitude resides in the range from millinewtons to micronewtons.

MEMS devices are often three-dimensional. MEMS packaging requires both electrical interconnection for signal transmission and mechanical interconnection for the interaction of the packaged device with its external environment [18.51, 67]. Many such mechanical interconnections require three-dimensional manipulation and three-dimensional force control. The actual operations are highly application specific, which pose great challenges to the development of automated microassembly systems. Automated IC packaging systems can be used for the packaging of certain MEMS devices such as accelerometers and gyros. However, packaging of mi-

crofluidic devices, optical MEMS devices, and hybrid microsystems often require the development of new automated microassembly techniques and systems.

The selection of assembly mode is among the first decisions to be made in developing automated microassembly systems. Sequential microassembly requires the use of micromanipulators and sensory feedback. At each moment, only one or a few parts are being assembled. Depending on the physical effects used, parallel microassembly can be either deterministic or stochastic [18.68]. Die bonding is an example of deterministic parallel microassembly. In stochastic parallel microassembly, large numbers of parts are assembled simultaneously using distributed physical effects such as electrostatic force, capillary force, centrifugal force, or vibration [18.60, 69–71]. In fact, the basic philosophy of stochastic parallel microassembly is to minimize the use of sensory feedback.

Each of these assembly modes has both advantages and disadvantages. Each has its own suitable applications. Deterministic parallel assembly shares several commonalities with sequential microassembly. For example, sensory feedback is often used in deterministic parallel microassembly. However, deterministic parallel microassembly requires high relative positioning accuracy between parts. In addition, only simple planar structure features can be assembled in order to make parallel operation possible.

Due to the requirement of MEMS packaging for three-dimensional manipulation and microassembly, it can be expected that automated sequential microassembly will be the most widely adopted solution. In particular, those MEMS packaging applications that require the control of multiple-degree-of-freedom (DOF) interaction force should use sequential microassembly. The major possible disadvantage of sequential microassembly is its low throughput. This constraint can often be overcome by appropriate system design.

Here we introduce a three-dimensional microassembly example that originates from an industrial appli-

cation [18.61]. It is significantly different from wire bonding and die bonding applications in that high-precision 3-D part insertion is required. Together with other examples, it will be used throughout this section to illustrate the major concepts and techniques and the logic connections between each functional unit of an automated microassembly system.

The assembly task is to pick up micromachined thin metal parts that are transferred to the assembly workcell on a vacuum-release tray (Fig. 18.6a, b and c), and insert them into vertically deep reactive ion etching (DRIE) etched holes in a silicon wafer (Fig. 18.6d). The wafers can have diameters up to 8 inches. The holes on each wafer form regular arrays consisting of approximately 50 holes each. However, these arrays may not be regularly distributed on the wafer. Typically, hundreds of parts are to be assembled on each wafer. In general, each assembly operation is a typical rectangular-peg-into-a-rectangular-hole problem. Each metal part is approximately half a millimeter in width and less than 100 μm in thickness at its rectangular tip. The total assembly tolerance is typically smaller than 10 μm in the vertical direction, and smaller than 20 μm in the horizontal direction. This task is a typical application of 3-D microassembly techniques.

18.6.2 Microassembly System Design

This section discusses the design of automated microassembly systems from the perspective of robotic systems and automation. Performance objects to be achieved include high reliability, high throughput, high flexibility, and low cost.

General Guidelines

Taking a system perspective. An automated microassembly system consists of many functional units and must integrate techniques from a diverse range of areas such as robotics, computer vision, microscope op-

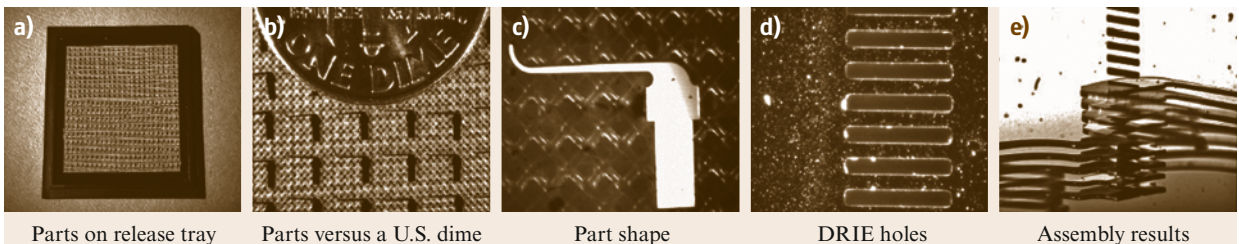


Fig. 18.6a–e A 3-D microassembly example

tics, physics, and chemistry. It is therefore important to consider the interaction between these units.

Emphasizing the coupling with packaging processes. Although in this section microassembly techniques are addressed primarily from the perspective of robotics for the clarity of presentation, robotic system designers must recognize the strong dependence of architecture design of the robotic systems on the packaging process being implemented. This connection should be emphasized from the beginning of system development.

Design for reconfigurability. Automated packaging machines must be designed so that they can be customized for a wide variety of applications. Reconfigurability is therefore a basic design requirement. Typically, modular design based on function decomposition is desirable. Support for tool replacement is essential.

The design of automated microassembly systems is strongly dependent on the assembly tolerance required. First of all, the repeatability of the motion control system and the micromanipulator used is determined by the required assembly tolerance. In addition, assembly tolerance often determines the minimum resolution of the microscope optics. General microassembly tasks may only require microscopic vision feedback and manipulators with microscale repeatability, while complex microassembly tasks may also require the integration of microforce and vision feedback [18.72, 73].

Automated microassembly systems must be able to support a wide variety of material-handling tools, which include part transfer tools, bulk feeders, wafer-handling tools, magazine loaders and unloaders, etc.

The role of the micromanipulator is to provide multiple-DOF fine motion control. The role of the microgripper is to grasp objects in pick-and-place and

other assembly operations. Their efficiency and robustness will to a great extent decide the performance of the entire system. Microgripper design is also closely related to the design of fixtures for microassembly operations.

Major environment factors include clean-room requirements, temperature, humidity, airflow, etc. Certain assembly operations must be performed in a clean room. This requires that the design of automated microassembly systems comply with relevant standards. Some packaging processes such as eutectic bonding must be performed under high temperature. Consequently, the potential influence of high temperature on motion control system and microscope optics must be considered. For the manipulation of microscale objects, environment conditions such as temperature and humidity can have a major influence on adhesion forces [18.74, 75]. Therefore, it is often important to consider environment control in system design.

Assembly Process Flow: An Example

The micromachined metal parts (Fig. 18.6c) are horizontally transferred to the workcell on the vacuum-release tray (Fig. 18.6b). The wafer is placed on a wafer mount perpendicular to the horizontal plane (Fig. 18.6b). This configuration does not require the flipping of the thin metal parts and is advantageous in terms of reliability and efficiency. There are two major operations in each assembly cycle: pickup and insertion. Under the configuration shown in Fig. 18.7a, all the operations are performed by the same workcell. Complex packaging operations must often be decomposed and performed by multiple workcells [18.76].

General System Architecture

An automated microassembly system typically consists of the following functional units.

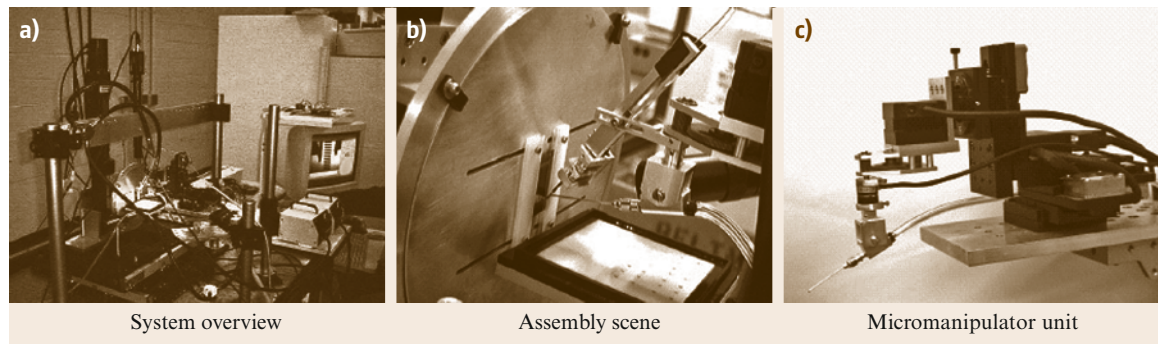


Fig. 18.7a–c An experimental microassembly workcell

Large workspace positioning unit. A large workspace and long-range positioning motion are required in most microassembly operations. The large workspace is necessary to accommodate different functional units, part feeders, and various tools.

In general, off-the-shelf motion control systems developed for automated IC packaging equipment can be adopted directly. For the task described in Sect. 18.6.1, the DRIE-etched holes are distributed on wafers up to 8 inches in diameter. This requires the assembly system to have a commensurate working space and high positioning speed. The coarse positioning unit has four DOFs (Fig. 18.7a). Planar motion in the horizontal direction is provided by an open-frame high-precision XY table with a travel of 32 cm (12 inch) and a repeatability of 1 μm in both directions. Position feedback with a resolution of 0.1 μm is provided by two linear encoders. A dual-loop PID plus feedforward control scheme is used for each axis. The internal speed loop is closed on the rotary encoder on the motor. The external position loop is closed on the linear encoder. Each wafer is placed on the vertical wafer mount that provides both linear and rotational control (Fig. 18.7b). Vertical motions of the wafer mount are provided by a linear slide with a travel of 20 cm (8 inch) and a repeatability of 5 μm . It is also controlled using a PID plus feedforward algorithm. Both the XY table and the vertical linear slides are actuated using alternating-current (AC) servo motors. The rotation of the wafer mount is actuated by an Oriental PK545AUA microstep motor with a maximum resolution of 0.0028°/step. All low-level controllers are commanded and coordinated by a host computer [18.61].

For applications requiring repeatability of one micron or greater, it is convenient to use conventional positioning tables to implement coarse range motions. These tables typically use leadscrew or ballscrew drives and ball or roller bearings. For applications requiring submicron or nanometer repeatability, a few solutions are also commercially available. For example, piezoactuators are often used for nanometer repeatability motion. The disadvantage of piezoactuators is that their travel range is small, typically on the order of 100 microns. As another example, a series of positioning stages with submicron repeatability based on direct-drive linear actuators and air bearings is available from Aerotech. It is also possible to use parallel structure mechanisms such as a Stewart platform [18.77]. In general, the development of IC manufacturing towards the deep submicron level provides a major driving force behind the development of these motion control techniques.

Micromanipulator unit. Fine pose (position and orientation) control is required in operations such as 3-D precision alignment and assembly. For example, the six DOFs required by the task introduced in Sect. 18.6.1 are implemented on separate structures. The three Cartesian DOF are provided by an adapted Sutter MP285 micromanipulator, which also provides yaw motions with its rotational DOF (Fig. 18.7c). Roll motions are implemented on the wafer mount (Fig. 18.7b). The pitch movement of the metal part after pickup is not motorized and is implemented through manual adjustment and calibration before assembly. More discussion of micromanipulator configuration can be found in [18.61].

Two principles need to be considered in implementing motion control for automated 3-D microassembly. The first is the partition of large-workspace coarse positioning unit and the fine positioning micromanipulator unit. The second is the decomposition and distributed implementation of multiple DOFs. In practice, the actual implementations of these principles are highly application specific. For certain applications, if a large-range positioning unit is sufficient to satisfy assembly tolerance requirements, the separate implementation of a micromanipulator may even be unnecessary.

In general, the separation of a micromanipulator unit will facilitate the implementation of high-bandwidth motion control. However, this separation normally brings redundancy to the entire motion control system. The functioning of the high-precision micromanipulator must also rely on closed-loop feedback control, especially microscopic vision feedback.

Automated microgripper unit. The function of a microgripper is to provide geometrical and physical constraints (grasping) in pick-and-place and assembly operations. The reliability and efficiency of the microgripper is critical to the performance of the entire automated microassembly system. Several factors must be considered in micromanipulator design. First of all, as the micromanipulator end-effector, the microgripper must be constantly monitored under a microscope. Therefore, it must be small in size and suitable in shape to remain in the microscope's field of view and to minimize occlusion. Secondly, it is important to consider the different governing physics and to explore the use of various gripping forces [18.77]. Thirdly, microgrippers are used in both part pick-and-place and assembly. Since assembly operations typically require more constraints, microgrippers designed for pick-and-place operations may not necessarily be suitable for assembly operations. In fact, the microassembly task described in Sect. 18.6.2

is performed using a combined microgripper [18.61]. Microgripper development is often closely related to the development of fixtures that can also have microscale sizes.

Microscope optics and imaging unit. The function of the microscope optics and imaging unit is to provide noncontact measurement of the geometry, motion, and spatial relations of assembly objects. Typical configurations of commercial device bonding systems use one or two vertical microscopes. An inverted microscope configuration is commonly used for backside alignment. On the other hand, 3-D microassembly may require two camera views in a stereo configuration. In the system shown in Fig. 18.7a, a total of four different views can be provided to its human operator: a global view of the entire assembly scene, a vertical microscopic view for part pickup, and two lateral microscopic views for the fine position and orientation adjustments during the final microassembly operations. Each view uses a CCD camera with a matching optical system. All images are captured using a Matrox Corona PCI frame grabber.

Microscopic visual feedback is crucial for precise 3-D alignment. Provided that resolution requirements of the assembly task can be satisfied, microscope optics with larger working distances is desirable. For the assembly task addressed in Sect. 18.6.1, an Edmund Scientific VZM 450i zoom microscope with a 1× objective is used to provide the right view. Its working distance is approximately 90 mm, with a resolution of 7.5 μm. The vertical view is also provided by a VZM 450i microscope with a 0.5× objective to guide pickup operations. Its field of view can range from 2.8×2.8 mm to 17.6×17.6 mm. Its working distance is approximately 147 mm.

If visual servoing is required in automate assembly operations, a stereo configuration formed by adding another lateral view may be necessary. Higher resolution may be necessary in some microassembly tasks. In such cases, the same configuration can be used with higher-resolution microscope optics, for example, two Navitar TenX zoom microscopes with Mitutoyo ultralong-working-distance M Plan Apo 10× objectives have been used in this configuration by the authors. The resolution of each microscope is 1 μm, with a working distance of 33.5 mm. Consequently, the usable workspace of the micromanipulator is reduced [18.61].

The global view is implemented using a miniature Marshall V-1260 board camera to monitor the status of the entire assembly scene. It plays an important role in helping the operator to understand gross spatial relations and preventing operation errors.

From the perspective of robotic systems, the development of automation microassembly systems for complex 3-D microassembly operation will rely on the development of:

1. Compact, robust and high-speed micromanipulators with 5–6 DOFs.
2. Highly reliable and efficient microgrippers that are suitable for working under microscopes. Such grippers should have active force control or passive compliance to avoid damage to MEMS devices.
3. Three-dimensional microscopic computer vision techniques, three-dimensional microforce measurement and control techniques, and their integration.

18.6.3 Basic Microassembly Techniques

This section introduces a few supporting techniques important to automated microassembly systems, including machine vision techniques, microforce control techniques, and simulation verification of assembly strategy.

Machine Vision Techniques

Machine vision techniques are widely used in the semiconductor industry. The major difference between machine vision and general computer vision [18.78] is that, unlike natural objects and scenes, industrial objects and scenes can often be artificially designed and configured. This advantage often makes it possible to significantly reduce the complexity and enhance the robustness of vision techniques. The applications of machine vision techniques can generally be categorized into the following two classes based on the requirement of real-time processing.

Non-time-critical vision applications. These applications do not require visual feedback for high-bandwidth real-time control. Examples include object recognition and packaging quality inspection [18.79].

Time-critical vision applications. These applications require real-time visual feedback. Examples include vision-guided pick-and-place, alignment, insertion, etc. [18.80]. An introduction to 3-D computer vision techniques can be found in [18.81]. A standard introduction to visual servoing techniques can be found in [18.82].

Several commercial software packages are available from suppliers such as Cognex, Coreco Imaging, and National Instruments.

Microforce Control Techniques

The theory of force control has been studied by the robotics community for more than 50 years [18.83, 84]. Several theoretical frameworks and many control algorithms have been proposed and experimentally verified. A variety of macroscale multiple-DOF force sensors have been developed.

Force control is also crucial to microassembly. For example, contact forces in device bonding must often be programmed and precisely controlled. In general, many of the macroscale force control techniques can be applied at micro/mesoscales. Force control for device bonding is essentially 1-D and involves force on the order of several newtons. On the other hand, in the manipulation of micro/mesoscale parts, the magnitude of interactive force typically ranges from millinewtons (10^{-3} N) to micronewtons (10^{-6} N). Forces of this magnitude are often referred to as *microforces*. A major technical challenge in implementing microforce control is the lack of multiple-DOF microforce sensors. A basic requirement for multiple-DOF microforce sensors is that they must be miniature in size. The manufacturing of these sensors normally requires the use of micromachining, including MEMS techniques. There are two major microforce sensing configurations.

Stand-alone force sensor. The advantage of this configuration is that the sensor is general purpose and can be used with different microgrippers. Most macroscale multiple-DOF force sensors are of this type. However, this also requires the sensor to have sufficient structural stiffness to support the static load of microgripper, which is often significantly larger than the force resolution to be reached.

Embedded force sensor. Micro strain gages can be attached to microgrippers [18.85]. Force-sensitive materials can also be deposited on microgrippers. This configuration avoids the issue of static load. However, such force sensing capabilities are dependent on the microgripper design, which often is not necessarily optimal for the measurement of multiple-DOF microforce/torque.

Complex and high-precision microassembly tasks also require the integration of microscopic machine vision with microforce control. Simple integration techniques use a gating/switching scheme [18.72, 86]. For more integrated approaches, visual impedance can be used [18.73].

Simulation Verification of Assembly Strategies

In many microassembly tasks, the distances between adjacent features are often on the meso/microscale. Therefore, selecting the correct assembly sequence is important for collision avoidance. In addition, due to the limited working distance of microscopes, microassembly operations must often be performed in a limited space using micromanipulators. Collision avoidance is critical to avoiding equipment or device damage. Potential collisions can be found and avoided by using offline simulation software. Many commercially available offline robot programming tools can provide this function.

Microassembly Tool

The end-effector of a micromanipulator in an automated microassembly system is often in the form of a microgripper, whose reliability and efficiency greatly influence the reliability and efficiency of the entire system. The microgripper must often be as small as possible. Its design must also minimize potential damage to fragile MEMS parts. This often requires passive compliance in structural design.

Microgrippers with integrated MEMS actuators can be fabricated monolithically and thus can be more compact in size. Several physical effects are commonly used in MEMS actuators, including electrostatic force and piezoelectric force [18.87], SMA (shape memory alloy) [18.88, 89], and thermal deformation [18.63, 90]. Currently, the major limitation is that it is difficult for MEMS actuators to generate sufficient travel, force, and power output. Alternatively, another solution is to provide actuation externally [18.61, 91]. The advantage is that sufficient travel and force and power output can be more easily obtained. The major disadvantage is that the microgripper is less compact in size. This could become a major obstacle to its applications.

18.7 Microrobotics

Today, more and more microrobotic devices are enabling new applications in various fields. Besides microassem-

bly, microrobotics can play important roles in other industry fields for manipulation, characterization, in-

spection, and maintenance, and in biotechnology for, e.g., manipulating cells, a field referred to as biomicrobotics.

18.7.1 Microrobots

Microrobotics is a field that combines the established theory and techniques of robotics with the exciting new tools provided by MEMS technology in order to create intelligent machines that operate at micron scales. As stated by authors reviewing the microrobotics field [18.31, 92], many *micro* terms such as *micromechanics*, *micromechanism*, *micromachines*, and *microrobots* are used synonymously to indicate a wide range of devices whose function is related to a *small* scale; however, *small* scale is a relative term so a clearer definition is needed.

The obvious difference between a macro-robot and a microrobot is the size of the robot. Thus, one definition of a microrobot is *a device having dimensions smaller than classical watch-making parts (i. e., μm to mm) and having the ability to move, apply forces and manipulate objects in a workspace with dimensions in the micrometer or submicrometer range* [18.94]. However, in many cases it is important that the robot can move over much larger distances. This task-specific definition

is quite wide and includes several types of very small robots as well as stationary micromanipulation systems, which are a few decimeters in size but can carry out very precise manipulation (in the micron or even nanometer range) [18.92].

Besides classification by task or size, microrobots can also be classified by their mobility and functionality [18.31, 93]. Many robots usually consist of sensors and actuators, a control unit, and an energy source.

Depending on the arrangement of these components, one can classify microrobots according to the following criteria: locomotive and positioning possibility (yes or no), manipulation possibility (yes or no), control type (wireless or tethered), and autonomy. Figure 18.8 illustrates 15 different possible microrobot configurations by combining the four criteria [18.31, 93].

As depicted in Fig. 18.8 (taken from [18.30]), the classification is dependent on the following microrobot components: the control unit (CU), the power source (PS), the actuators necessary for moving the robot platform (i. e., the robot drive for locomotion and positioning; AP), and the actuators necessary for operation (i. e., manipulation using robot arms and hands; AM). Besides the different actuation functions, sensory functions are also needed, for example, tactile sensors for microgrippers or charge-coupled device (CCD) cam-

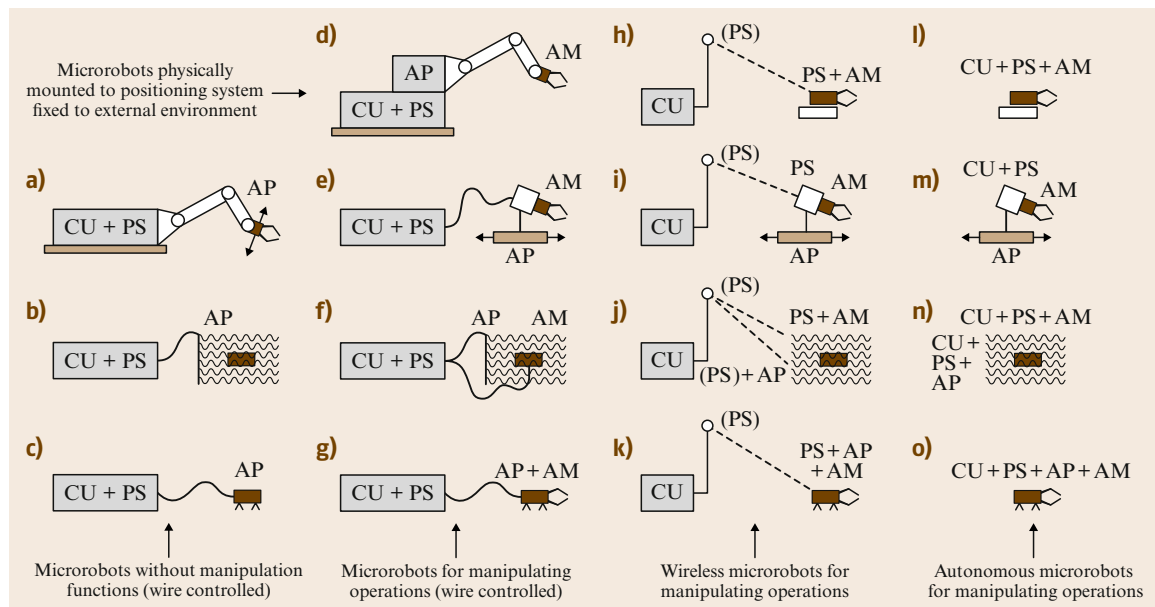


Fig. 18.8a–o Classification of microrobots by functionality (modification of earlier presented classification schemes [18.12, 31, 92, 93]). (CU indicates the control unit; PS, the power source or power supply; AP, the actuators for positioning; AM, the actuators for manipulation)

eras for endoscopic applications (compare Fig. 18.8d and a).

The ultimate goal is to create a fully autonomous, wireless mobile microrobot equipped with suitable microtools according to Fig. 18.8o. Because this is a very difficult task, a good start is to investigate the possibility of making silicon microrobot platforms that are steered and powered through wires, like the one in Fig. 18.8c, and to study their locomotion capability.

The majority of MEMS-based microrobotic devices developed so far could be categorized as moveable links – microcatheters [18.95, 96], according to Fig. 18.8a, or microgrippers [18.24], such as those in Fig. 18.8d, or the microgrippers [18.97, 98] shown in Fig. 18.8e. Among the research publications covering locomotive microrobots, most publications have addressed microconveyance systems (Fig. 18.8b) [18.99–102]. Robots using external sources for locomotion could be used (compare Fig. 18.8b,f,j,n). According to *Fatikow* and *Rembold* [18.92], several researchers are working on methods to navigate micromechanisms through human blood vessels; however, these microrobots are difficult to control. Examples of partially autonomous systems (compare Fig. 18.8j) are the concept for so-called *smart pills*. Centimeter-sized pills for sensing temperature and/or pH inside the body have been presented [18.103, 104] as well as pills equipped with video cameras [18.105]. The pill is swallowed and transported to the part of the body where one wants to measure or record a video sequence. The information of the measured parameter or the signals from the camera is then transmitted (telemetrically) out of the body. More sophisticated approaches involving actuators for drug delivery of various kinds have also been proposed [18.92, 104]. The position of the pill inside the body is located by an X-ray monitor or ultrasound. As soon as the pill reaches an infected area, a drug encapsulated in the pill can be released by the actuators onboard. External communication could be realized through radio signals.

Several important results have been presented regarding walking microrobots (Fig. 18.8c,g,k) fabricated by MEMS technologies and batch manufacturing. Different approaches for surface-micromachined robots [18.106, 107] and for a piezoelectric dry-reactive-ion-etched microrobot should be mentioned. A suitable low-power application-specific integrated circuit (ASIC) for robot control has been successfully tested and is planned to be integrated on a walking microrobot [18.108]. The large European Esprit project *MINIMAN* (1997) has the goal of developing

moveable microrobotic platforms with integrated tools with six degrees of freedom for applications such as microassembly within an SEM and involves different MEMS research groups from several universities and companies across Europe. Further, miniature robot systems with MEMS/MST (MicroSystem Technology) components have been developed [18.109].

Several research publications on *gnat* minirobots and actuator technologies for MEMS microrobots [18.3, 110] were reported by US researchers in the early 1990s, and several groups in Japan are also currently developing miniaturized robots based on MEMS devices [18.8]. In Japan, an extensive ten-year program on *micromachine technology*, supported by the Ministry of International Trade and Industry (MITI), started in 1991. One of the goals of this project is to create micro-sized and miniature robots for microfactory, medical technology, and maintenance applications. Several microrobotic devices, including locomotive robots and microconveyers, have been produced within this program. Miniature robot devices or vehicles [18.111] for locomotive tasks, containing several MEMS components, have been presented. Even though great efforts have been made on robot miniaturization using MEMS technologies, no experimental results on MEMS batch-fabricated microrobots suitable for autonomous walking (i.e., robust enough to be able to carry its own power source or to be powered by telemetric means) have been presented yet. The first batch-fabricated MEMS-based microrobot platform able to walk was presented in 1999 [18.112]. However, this robot was powered through wires and was not equipped with manipulation actuators. Besides walking microrobotic devices, several reports on flying [18.113–115] and swimming [18.116] robots have been published. Micromotors and gear boxes made using LIGA technology (a high-precision, lithographically defined plating technology) are used to build small flying microhelicopters, which are commercially available from the Institute of Microtechnology in Mainz, Germany, as rather expensive demonstration objects [18.114]. Besides the pure mechanical microrobots, hybrid systems consisting of electromechanical components and living organisms such as cockroaches have also been reported [18.117].

18.7.2 Bio-microrobotics

Biomanipulation entails such operations as positioning, grasping, and injecting material into various locations in cells. Research topics in bio-microrobotics include the autonomous manipulation of single cells or molecules,

the characterization of biomembrane mechanical properties using microrobotic systems with integrated vision and force sensing modules, and more. The objective is to obtain a fundamental understanding of single-cell biological systems and provide characterized mechanical models of biomembranes for deformable cell tracking during biomanipulation and cell injury studies.

Existing biomanipulation techniques can be classified into noncontact manipulation including laser trapping [18.118–121] and electrorotation [18.122–124], and contact manipulation, referred to as mechanical micromanipulation [18.125]. When laser trapping [18.118–121] is used for noncontact biomanipulation, a laser beam is focused through a large-numerical-aperture objective lens, converging to form an optical trap in which the lateral trapping force moves a cell in suspension toward the center of the beam. The

longitudinal trapping force moves the cell in the direction of the focal point. The optical trap levitates the cell and holds it in position. Laser traps can work in a well-controlled manner. However, two features make laser trapping techniques undesirable for automated cell injection. The high dissipation of visible light in aqueous solutions requires the use of high-energy light close to the UV spectrum, raising the possibility of damage to the cell. Even though some researchers claim that such concerns could be overcome using wavelengths in the near-infrared (IR) spectrum [18.120], the question as to whether the incident laser beam might induce abnormalities in the cells' genetic material still exists. One alternative to using laser beams is the electrorotation technique. Electric-field-induced rotation of cells was demonstrated by *Mischel* [18.126], *Arnold* [18.127], and *Washizu* [18.124]. This noncontact cell manipulation technique is based on controlling the phase shift and magnitude of electric fields. These fields, appropriately applied, produce a torque on the cell. Different system configurations have been established for cell manipulation based on this principle [18.122, 123], which can achieve high accuracy in cell positioning. However, it lacks a means to hold the cell in place for further manipulation, such as injection, since the magnitude of the electric fields has to be kept low to ensure the viability of cells. The limits of noncontact biomanipulation in the laser trapping and electrorotation techniques make mechanical micromanipulation desirable. The damage caused by laser beams in the laser trapping technique and the lack of a holding mechanism in the electrorotation technique can be overcome by mechanical micromanipulation.

To improve the low success rate of manual operation, and to eliminate contamination, an autonomous robotic system (shown in Fig. 18.9) has been developed to deposit DNA into one of the two nuclei of a mouse embryo without inducing cell lysis [18.11, 128]. The laboratory's experimental results show that the success rate for the autonomous embryo pronuclei DNA injection is dramatically improved over conventional manual injection methods. The autonomous robotic system features a hybrid controller that combines visual servoing and precision position control, pattern recognition for detecting nuclei, and a precise autofocus scheme. Figure 18.10 illustrates the injection process.

To realize large-scale injection operations, a MEMS cell holder was fabricated using anodic wafer-bonding techniques. Arrays of holes are aligned on the cell holder, which are used to contain and fix individual cells for

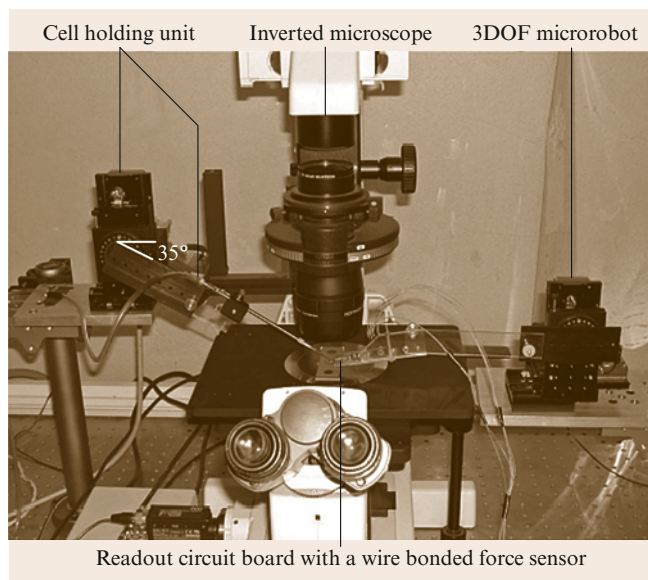


Fig. 18.9 Robotic biomanipulation system with vision and force feedback

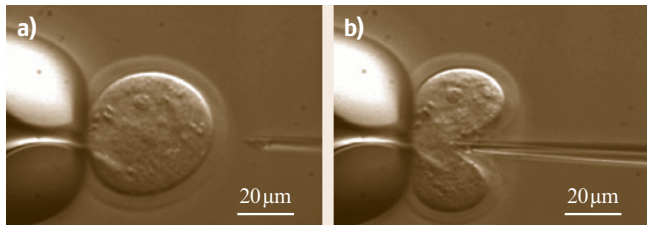


Fig. 18.10a,b Cell injection process. (a)(b) Mouse oocyte zona pellucida (ZP)

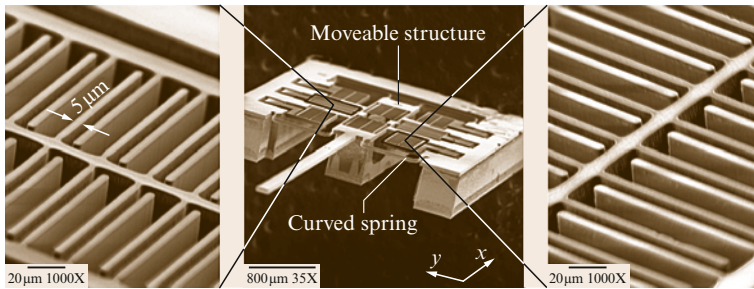


Fig. 18.11 A cellular force sensor with orthogonal comb drives detailed

injection. When well calibrated, the system with the cell holder makes it possible to inject large numbers of cells using position control. The cell injection operation can be conducted in a move–inject–move manner.

A successful injection is determined greatly by injection speed and trajectory, and the forces applied to cells. To further improve the robotic system's performance, a multi-axial MEMS-based capacitive cellular force sensor is being designed and fabricated to provide realtime force feedback to the robotic system. The MEMS cellular force sensor also aids research in biomembrane mechanical property characterization.

MEMS-Based Multi-Axis Capacitive Cellular Force Sensor

The MEMS-based two-axis cellular force sensor [18.129] shown in Fig. 18.11 is capable of resolving normal forces applied to a cell as well as tangential forces generated by improperly aligned cell probes. A high-yield microfabrication process was developed to form the 3-D high-aspect-ratio structure using deep reactive ion etching (DRIE) on silicon-on-insulator (SOI) wafers. The constrained outer frame and the inner movable structure are connected by four curved springs. A load applied to the probe causes the inner structure to move, changing the gap between each pair of interdigitated comb capacitors. Consequently, the total capacitance change resolves the applied force. The interdigitated capacitors are orthogonally configured to make the force sensor capable of resolving forces in both the x - and y -directions. The cellular force sensors used in the experiments are

capable of resolving forces up to $25\ \mu\text{N}$ with a resolution of $0.01\ \mu\text{N}$.

Tip geometry affects the quantitative force measurement results. A standard injection pipette (Cook K-MPIP-1000-5) tip section with a tip diameter of $5\ \mu\text{m}$ is attached to the probe of the cellular force sensors.

The robotic system and high-sensitivity cellular force sensor are also applied to biomembrane mechanical property studies [18.130]. The goal is to obtain a general parameterized model describing cell membrane deformation behavior when an external load is applied. This parameterized model serves two chief purposes. First, in robotic biomanipulation, it allows online parameter recognition so that cell membrane deformation behavior can be predicted. Second, for a thermodynamic model of membrane damage in cell injury and recovery studies, it is important to appreciate the mechanical behavior of the membranes. This allows the interpretation of such reported phenomena as mechanical resistance to cellular volume reduction during dehydration, and its relationship to injury. The establishment of such a biomembrane model will greatly facilitate cell injury studies.

Experiments demonstrate that robotics and MEMS technology can play important roles in biological studies such as automating biomanipulation tasks. Aided by robotics, the integration of vision and force sensing modules, and MEMS design and fabrication techniques, investigations are being conducted in biomembrane mechanical property modeling, deformable cell tracking, and single-cell and biomolecule manipulation.

18.8 Nanorobotics

Nanorobotics represents the next stage in miniaturization for maneuvering nanoscale objects. Nanorobotics is the study of robotics at the nanometer scale, and includes robots that are nanoscale in size, i. e., nanorobots,

and large robots capable of manipulating objects that have dimensions in the nanoscale range with nanometer resolution, i. e., nanorobotic manipulators. Robotic manipulation at nanometer scales is a promising tech-

nology for structuring, characterizing, and assembling nanoscale building blocks into NEMS. Combined with recently developed nanofabrication processes, a hybrid approach is realized to build NEMS and other nanorobotic devices from individual carbon nanotubes and SiGe/Si nanocoils. Material science, biotechnology, electronics, and mechanical sensing and actuation will benefit from advances in nanorobotics.

18.8.1 Introduction to Nanomanipulation

Nanomanipulation, or positional and/or force control at the nanometer scale, is a key enabling technology for nanotechnology by filling the gap between top-down and bottom-up strategies, and may lead to the appearance of replication-based molecular assemblers [18.18]. These types of assemblers have been proposed as general-purpose manufacturing devices for building a wide range of useful products as well as copies of themselves (self-replication).

Presently, nanomanipulation can be applied to the scientific exploration of mesoscopic physical phenomena, biology, and the construction of prototype nanodevices. It is a fundamental technology for property characterization of nanomaterials, nanostructures, and nanomechanisms, for the preparation of nanoscale building blocks, and for the assembly of nanodevices such as NEMS.

Nanomanipulation was enabled by the inventions of the STM [18.19], AFMs [18.44], and other types of SPMs. Besides these, optical tweezers (laser trapping) [18.131] and magnetic tweezers [18.132] are also

potential nanomanipulators. Nanorobotic manipulators (NRMs) [18.133, 134] are characterized by the capability of 3-D positioning, orientation control, independently actuated multiple end-effectors, and independent real-time observation systems, and can be integrated with scanning probe microscopes. NRMs largely extend the complexity of nanomanipulation.

A concise comparison of STM, AFM, and NRM technology is shown in Fig. 18.12. With its incomparable imaging resolution, an STM can be applied to particles as small as atoms with atomic resolution. However, limited by its 2-D positioning and available strategies for manipulations, standard STMs are ill-suited for complex manipulation and cannot be used in 3-D space. An AFM is another important type of nanomanipulator. Manipulation with an AFM can be done in either contact or dynamic mode. Generally, manipulation with an AFM involves moving an object by touching it with a tip. A typical manipulation starts by imaging a particle in noncontact mode, then removing the tip oscillation voltage and sweeping the tip across the particle in contact with the surface and with the feedback disabled. Mechanical pushing can exert larger forces on objects and, hence, can be applied for the manipulation of relatively larger objects. 1- to 3-D objects can be manipulated on a 2-D substrate. However, the manipulation of individual atoms with an AFM remains a challenge. By separating the imaging and manipulation functions, nanorobotic manipulators can have many more degrees of freedom including rotation for orientation control, and hence can be used for the manipulation of 0-D (symmetric spheres) to 3-D objects in 3-D free space. Limited by the lower resolution of electron microscopes, NRMs are difficult to use for the manipulation of atoms. However, their general robotic capabilities, including 3-D positioning, orientation control, independently actuated multiple endeffectors, separate real-time observation system, and integration with SPMs inside, make NRMs quite promising for complex nanomanipulation.

The first nanomanipulation experiment was performed by Eigler and Schweizer in 1990 [18.135]. They used an STM and materials at low temperatures (4 K) to position individual xenon atoms on a single-crystal nickel surface with atomic precision. The manipulation enabled them to fabricate rudimentary structures of their own design, atom by atom. The result is the famous set of images showing how 35 atoms were moved to form the three-letter logo IBM, demonstrating that matter could indeed be maneuvered atom by atom as Feynman suggested [18.1].

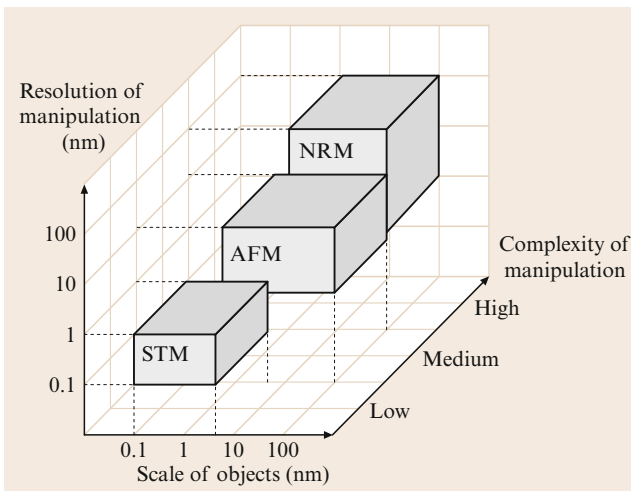


Fig. 18.12 Comparison of nanomanipulators

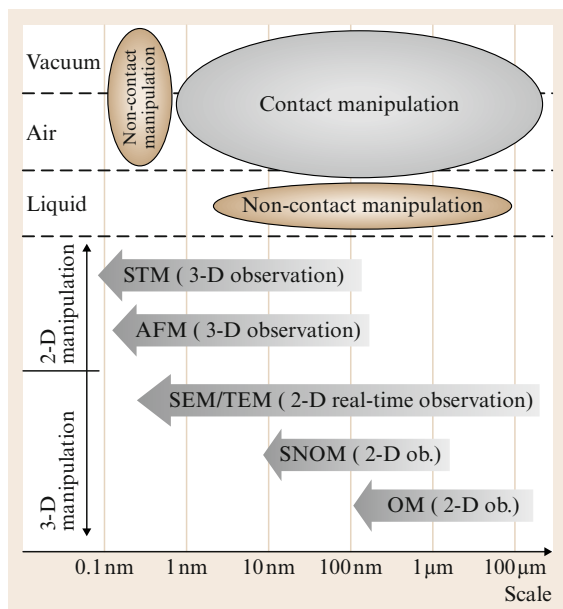


Fig. 18.13 Microscopes, environments, and strategies of nanomanipulation

A nanomanipulation system generally includes nanomanipulators as the positioning device, microscopes as *eyes*, various end-effectors including probes and tweezers among others as its *fingers*, and various types of sensors (force, displacement, tactile, strain, etc.) to facilitate the manipulation and/or to determine the properties of the objects. Key technologies for nanomanipulation include observation, actuation, measurement, system design and fabrication, calibration and control, communication, and the human-machine interface.

Strategies for nanomanipulation are basically determined by the environment – air, liquid or vacuum – which is further decided by the properties and size of the objects and observation methods. Figure 18.13 depicts the microscopes, environments, and strategies of nanomanipulation. In order to observe manipulated objects, STMs can provide sub-Ångström imaging resolution, whereas AFMs can provide atomic resolutions. Both can obtain 3-D surface topology. Because AFMs can be used in an ambient environment, they provide a powerful tool for biomanipulation that may require a liquid environment. The resolution of SEM is limited to about 1 nm, whereas field-emission SEM (FESEM) can achieve higher resolutions. SEM/FESEM can be used for real-time 2-D observation for both the objects and end-effectors of manipulators, and large ultrahigh-vacuum

(UHV) sample chambers provide enough space to contain an NRM with many degrees of freedom (DOFs) for 3-D nanomanipulation. However, the 2-D nature of the observation makes positioning along the electron-beam direction difficult. High-resolution transmission electron microscopes (HRTEM) can provide atomic resolution. However, the narrow UHV specimen chamber makes it difficult to incorporate large manipulators. In principle, optical microscopes (OMs) cannot be used for nanoscale (smaller than the wavelength of visible lights) observation because of diffraction limits. Scanning near-field OMs (SNOMs) break this limitation and are promising as a real-time observation device for nanomanipulation, especially for ambient environments. SNOMs can be combined with AFMs, and potentially with NRMs for nanoscale biomanipulation.

Nanomanipulation processes can be broadly classified into three types:

1. lateral noncontact
2. lateral contact
3. vertical manipulation

Generally, lateral noncontact nanomanipulation is mainly applied for atoms and molecules in UHV with an STM or bio-object in liquid using optical or magnetic tweezers. Contact nanomanipulation can be used in almost any environment, generally with an AFM, but is difficult for atomic manipulation. Vertical manipulation can be performed by NRMs. Figure 18.14 shows the processes of the three basic strategies.

Motion of the lateral noncontact manipulation processes is shown in Fig. 18.14a. Applicable effects [18.136] able to cause the motion include long-range van der Waals (vdW) forces (attractive) generated by the proximity of the tip to the sample [18.137], electric-field-induced fields by the voltage bias between the tip and the sample [18.138, 139], tunneling current local heating or inelastic tunneling vibration [18.140, 141]. With these methods, some nanodevices and molecules have been assembled [18.142, 143]. Laser trapping (optical tweezers) and magnetic tweezers are possible for noncontact manipulation of nanoscale biosamples, e.g., DNA [18.144, 145].

Noncontact manipulation combined with STMs has revealed many possible strategies for manipulating atoms and molecules. However, for the manipulation of carbon nanotubes (CNTs) no examples have been demonstrated.

Pushing or pulling nanometer objects on a surface with an AFM is a typical manipulation using this method as shown in Fig. 18.14b. Early work showed

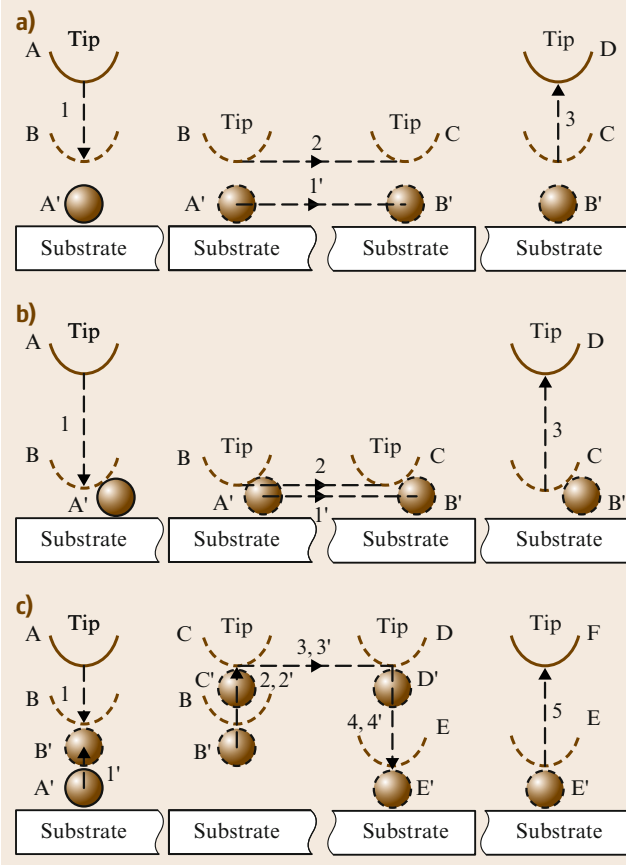


Fig. 18.14a–c Basic strategies of nanomanipulation. In the figure, A, B, C, ... represent the positions of end-effector (e.g., a tip); A', B', C', ... the positions of objects; 1, 2, 3, ... the motions of end-effector; and 1', 2', 3', ... the motions of objects. Tweezers can be used in pick-and-place to facilitate the picking-up, but are generally not necessarily helpful for placing. **(a)** Lateral noncontact nanomanipulation (sliding). **(b)** Lateral contact nanomanipulation (pushing/pulling). **(c)** Vertical nanomanipulation (picking and placing)

the effectiveness of this method for the manipulation of nanoparticles [18.146–150]. This method has also been shown in nanoconstruction [18.151] and biomanipulation [18.152]. A virtual-reality interface facilitates such manipulation [18.153–155] and may create an opportunity for other types of manipulation. This technique has been used in the manipulation of nanotubes on a surface, and some examples will be introduced later in this chapter.

The pick-and-place task as shown in Fig. 18.14c is especially significant for 3-D nanomanipulation since

its main purpose is to assemble prefabricated building blocks into devices. The main difficulty is in achieving sufficient control of the interaction between the tool and object and between the object and the substrate. Two strategies have been presented for micromanipulation [18.156] and have also proven to be effective for nanomanipulation [18.134]. One strategy is to apply a dielectrophoretic force between a tool and an object as a controllable additional external force by applying a bias between the tool and the substrate on which the object is placed. Another strategy is to modify the van der Waals and other intermolecular and surface forces between the object and the substrate. For the former, an AFM cantilever is ideal as one electrode to generate a nonuniform electrical field between the cantilever and the substrate.

18.8.2 Nanorobotic Manipulation Systems

Nanorobotic manipulators are the core components of nanorobotic manipulation systems. The basic requirements for a nanorobotic manipulation system for 3-D manipulation include nanoscale positioning resolution, a relative large working space, enough DOFs including rotational ones for 3-D positioning and orientation control of the end-effectors, and usually multiple end effectors for complex operations.

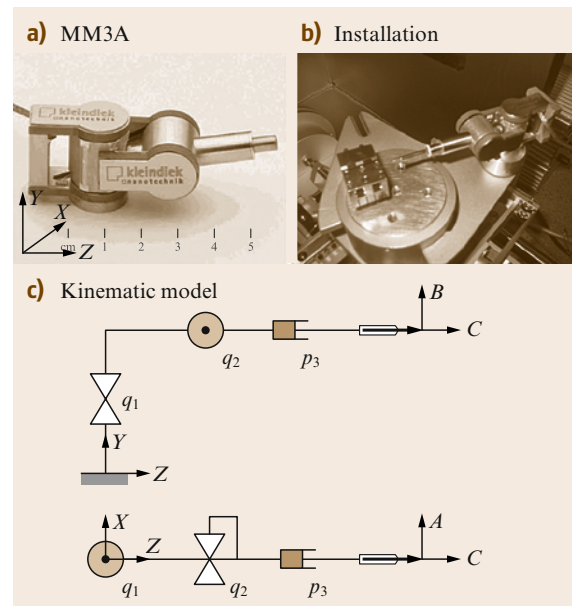


Fig. 18.15 **(a)** Nanomanipulator (MM3A)TM from Kleindiek **(b)** inside an SEM

Table 18.4 Specifications of MM3A

Item	Specification
Operating range q_1 and q_2	240°
Operating range Z	12 mm
Resolution A (horiz.)	10^{-7} rad (5 nm)
Resolution B (vert.)	10^{-7} rad (3.5 nm)
Resolution C (linear)	0.25 nm
Fine (scan) range A	20 μ m
Fine (scan) range B	15 μ m
Fine (scan) range C	1 μ m
Speed A, B	10 mm/s
Speed C	2 mm/s

A commercially available nanomanipulator (MM3ATM from Kleindiek) installed inside an SEM (Carl Zeiss DSM962) is shown in Fig. 18.15. The manipulator has three degrees of freedom, and nanometer-

to subnanometer-scale resolution (Table 18.4). Calculations show that when moving/scanning in the A/B direction at joint q_1/q_2 , the additional linear motion in C is very small. For example, when the arm length is 50 mm, the additional motion in the C direction is only 0.25 nm to 1 nm when moving 5–10 μ m; in the A direction; these errors can be ignored or compensated with an additional motion of the prismatic joint p_3 , which has a 0.25 nm resolution.

Figure 18.16a shows a nanorobotic manipulation system that has 16 DOFs in total and can be equipped with three or four AFM cantilevers as end-effectors for both manipulation and measurement. The positioning resolution is sub-nanometer and strokes are on the order of centimeters. The manipulation system is not only for nanomanipulation, but also for nanoassembly, nanoinstrumentation, and nanofabrication. Four-probe semiconductor measurements are perhaps the most complex manipulation that this system can perform, because it is necessary to actuate four probes independently by using four manipulators. With the advancement of nanotechnology, one could shrink the size of nanomanipulators and insert more DOFs inside the limited vacuum chamber of a microscope, and perhaps the molecular version of manipulators such as that dreamed of by Drexler could be realized [18.18].

For the construction of multi-walled carbon nanotube (MWNT)-based nanostructures, manipulators position and orient nanotubes for the fabrication of nanotube probes and emitters, for performing nanosoldering with electron-beam-induced deposition (EBID) [18.157], for the property characterization of single nanotubes for selection purposes and for characterizing junctions to test connection strength.

A nanolaboratory is shown in Fig. 18.16b. The nanolaboratory integrates a nanorobotic manipulation system with a nanoanalytical system and a nanofabrication system, and can be applied for manipulating nanomaterials, fabricating nanoscale building blocks, assembling nanodevices, and for in situ analysis of the properties of such materials, building blocks, and devices. Nanorobotic manipulation within the nanolaboratory has opened a new path for constructing nanosystems in 3-D space, and will create opportunities for new nanoinstrumentation and nanofabrication processes.

18.8.3 Nanorobotic Assembly

Nanomanipulation is a promising strategy for nanoassembly. Key techniques for nanoassembly include the structuring and characterization of nanoscale building

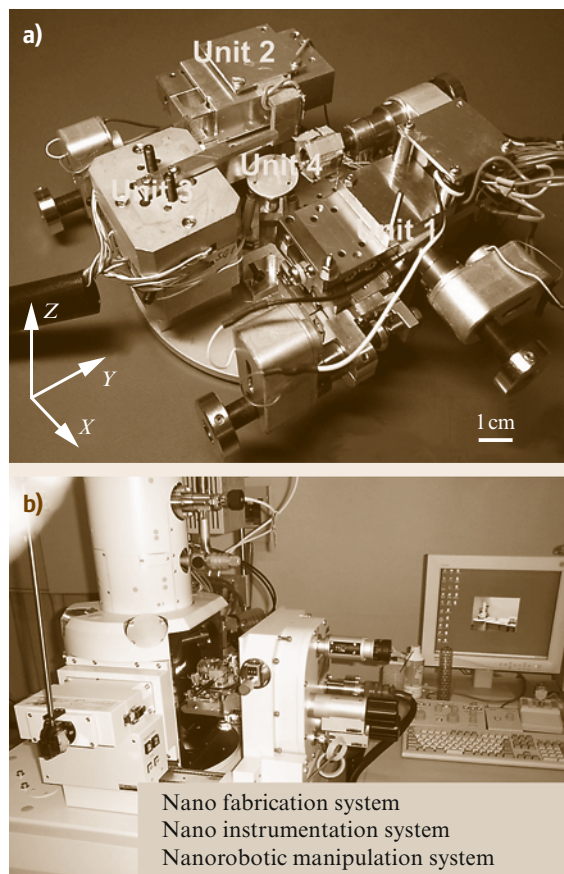


Fig. 18.16a,b Nanorobotic system. (a) Nanorobotic manipulators. (b) System setup

blocks, the positioning and orientation control of the building blocks with nanoscale resolution, and effective connection techniques. Nanorobotic manipulation, which is characterized by multiple DOFs with both position and orientation controls, independently actuated multiprobes, and a real-time observation system, have been shown effective for assembling nanotube-based devices in 3-D space.

The well-defined geometries, exceptional mechanical properties, and extraordinary electric characteristics, among other outstanding physical properties of CNTs [18.158] qualify them for many potential applications, especially in nanoelectronics [18.159], NEMS, and other nanodevices [18.160]. For NEMS, some of the most important characteristics of nanotubes include their nanometer diameter, large aspect ratio (10-1000), TPa-scale Young's modulus [18.161–163], excellent elasticity [18.133], ultralow interlayer friction, excellent capability for field emission, various electric conductivities [18.164], high thermal conductivity [18.165], high current-carrying capability with essentially no heating [18.166], sensitivity of conductance to various physical or chemical changes, and charge-induced bond-length change.

Helical 3-D nanostructures, or nanocoils, have been synthesized from various materials, including

helical carbon nanotubes [18.167] and zinc oxide nanobelts [18.168]. A new method of creating structures with nanoscale dimensions has recently been presented [18.169] and can be fabricated in a controllable way [18.170, 171]. The structures are created through a top-down fabrication process in which a strained nanometer-thick heteroepitaxial bilayer curls up to form 3-D structures with nanoscale features. Helical geometries and tubes with diameters between 10 nm and 10 μm have been achieved. Because of their interesting morphology, mechanical, electrical, and electromagnetic properties, potential applications of these nanostructures in NEMS include nanosprings [18.172], electromechanical sensors [18.173], magnetic field detectors, chemical or biological sensors, generators of magnetic beams, inductors, actuators, and high-performance electromagnetic wave absorbers.

NEMS based on individual carbon nanotubes and nanocoils are of increasing interest, indicating that capabilities for incorporating these individual building blocks at specific locations on a device must be developed. Random spreading [18.174], direct growth [18.175], self-assembly [18.176], dielectrophoretic assembly [18.177, 178], and nanomanipulation [18.179] have been demonstrated for positioning as-grown nanotubes on electrodes for the construction of these devices. However, for nanotube-based structures, nanorobotic assembly is still the only technique capable of in situ structuring, characterization, and assembly. Because the as-fabricated nanocoils are not free-standing from their substrate, nanorobotic assembly is virtually the only way to incorporate them into devices at present.

Nanorobotic Assembly of Carbon Nanotubes

Nanotube manipulation in two dimensions on a surface was first performed with an AFM by contact pushing on a substrate. Figure 18.17 shows the typical methods for 2-D pushing. Although similar to that shown in Fig. 18.14b, the same manipulation caused various results because nanotubes cannot be regarded as a 0-D point. The first demonstration was given by Lieber and coworkers for measuring the mechanical properties of a nanotube [18.180]. They adopt the method shown in Fig. 18.17b, i. e., to bend a nanotube by pushing one end of it and fixing the other end. The same strategy was used for the investigation of the behavior of nanotubes under large strain [18.181]. Dekker and coworkers applied the strategies shown in Fig. 18.17c,d to obtain a kinked junction and crossed nanotubes [18.182]. Avouris and coworkers combined this technique with an inverse process, namely straightening by pushing

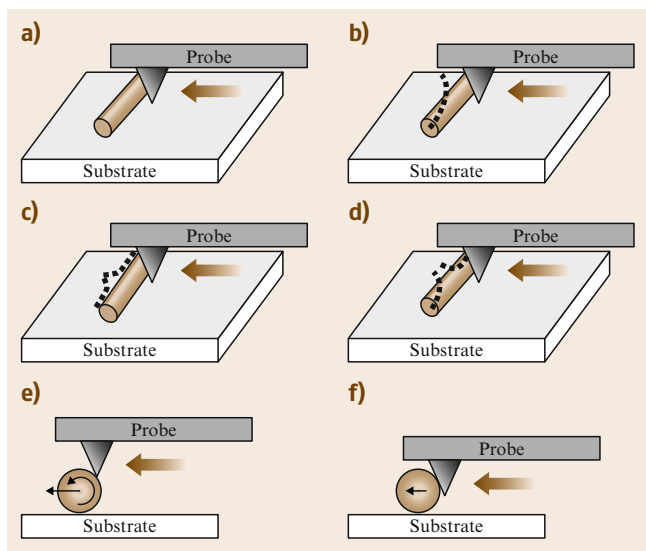


Fig. 18.17a–f Two-dimensional manipulation of CNTs. Starting from the original state shown in (a), pushing the tube at different site with different force may cause the tube to deform as in (b) and (c), to break as in (d), or to move as in (e) and (f). (a)Original state. (b)Bending. (c)Kinking. (d)Breaking. (e)Rolling. (f)Sliding

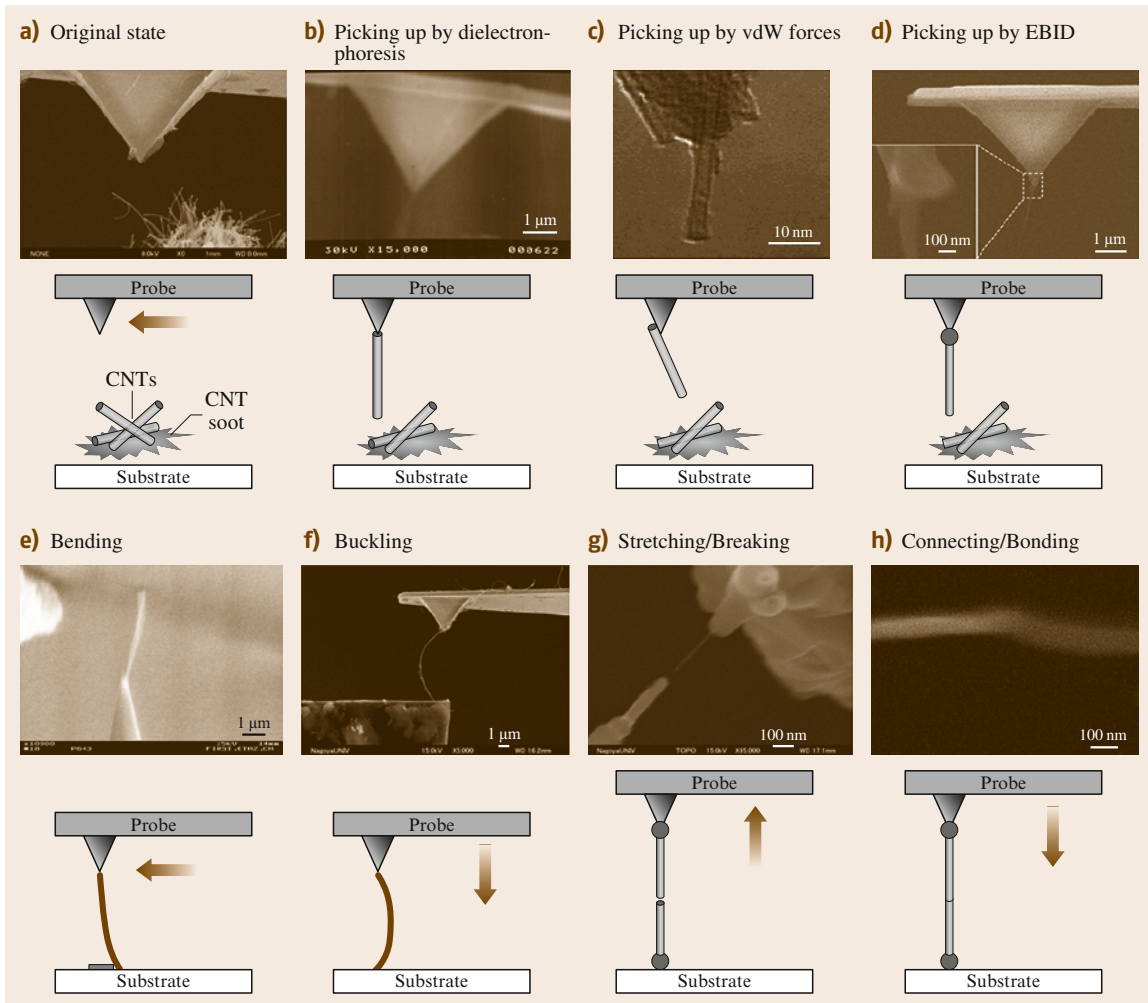


Fig. 18.18a–h Nanorobotic manipulation of CNTs. The basic technique is to pick up an individual tube from CNT soot (as in (a)) or from an oriented array; (b) shows a free-standing nanotube picked up by dielectrophoresis generated by a nonuniform electric field between the probe and substrate, (c) (after [18.188]) and (d) show the same manipulation by contacting a tube with the probe surface or fixing (e.g., with EBID) a tube to the tip (inset shows the EBID deposit). Vertical manipulation of nanotubes includes bending (e), buckling (f), stretching/breaking (g), and connecting/bonding (h). All examples with the exception of (c) are from the authors' work

along a bent tube, and realized the translation of the tube to another location [18.183] and between two electrodes to measure the conductivity [18.184]. This technique was also used to place a tube on another tube to form a single-electron transistor (SET) with a cross-junction of nanotubes [18.185]. Pushing-induced breaking (Fig. 18.17d) has also been demonstrated for a nanotube [18.183]. The simple assembly of two bent tubes and a straight one formed a Greek letter θ . To inves-

tigate the dynamics of rolling at the atomic level, rolling and sliding of a nanotube (as shown in Fig. 18.17e,f) are performed on graphite surfaces using an AFM [18.186]. Besides pushing and pulling, another important process is indentation. By indenting a surface, mechanical property characterization [18.187] and data storage [18.188] can be realized.

Manipulation of CNTs in 3-D space is important for assembling CNTs into structures and devices. Basic

techniques for the nanorobotic manipulation of carbon nanotubes are shown in Fig. 18.18 [18.189]. These serve as the basis for handling, structuring, characterizing, and assembling NEMS.

The basic procedure is to pick up a single tube from nanotube soot (Fig. 18.18a). This has been shown first by using dielectrophoresis [18.134] through nanorobotic manipulation (Fig. 18.18b). By applying a bias between a sharp tip and a plane substrate, a nonuniform electric field can be generated between the tip and the substrate with the strongest field near the tip. This field can cause a tube to orient along the field or further *jump* to the tip by electrophoresis or dielectrophoresis (determined by the conductivity of objective tubes). Removing the bias, the tube can be placed at other locations at will. This method can be used for free-standing tubes on nanotube soot or on a rough surface on which surface van der Waals forces are generally weak. A tube strongly rooted in CNT soot or lying on a flat surface cannot be picked up in this way. The interaction between a tube and the atomic flat surface of AFM cantilever tip has been shown to be strong enough for picking up a tube onto the tip [18.190] (Fig. 18.18c). By using EBID, it is possible to pick up and fix a nanotube onto a probe [18.191] (Fig. 18.18d). For handling a tube, weak connection between the tube and the probe is desired.

Bending and buckling a CNT as shown in Fig. 18.18e,f are important for in situ property characterization of a nanotube [18.192], which is a simple way to obtain the Young's modulus of a nanotube without damaging the tube (if performed within its elastic range) and hence can be used for the selection of a tube with desired properties. By buckling an MWNT over its elastic limit, a kinked structure can be obtained [18.193]. To obtain any desired angle for a kinked junction it is possible to fix the shape of a buckled nanotube within its elastic limit by using EBID [18.193]. For a CNT, the maximum angular displacement will appear at the fixed left end under pure bending or at the middle point under pure buckling. A combination of these two kinds of loads will achieve a controllable position of the kinked point and a desired kink angle. If the deformation is within the elastic limit of the nanotube, it will recover as the load is released. To avoid this, EBID can be applied at the kinked point to fix the shape.

Stretching a nanotube between two probes or a probe and a substrate has generated several interesting results (Fig. 18.18g). The first demonstration of 3-D nanomanipulation of nanotubes took this as an example to show the breaking mechanism and to measure the tensile strength of CNTs [18.133]. By breaking an MWNT in

a controlled manner, interesting nanodevices have been fabricated. This technique – destructive fabrication – has been presented to get sharpened and layered structures of nanotubes, and to improve the length control of a nanotube [18.193]. Typically, a layered and a sharpened structure can be obtained from this process, similar to that achieved from electric pulses [18.194]. Bearing motion has also been observed in an incompletely broken MWNT [18.193]. The interlayer friction has been shown to be very small [18.195, 196].

The reverse process, namely the connection of broken tubes (Fig. 18.18h), has been demonstrated recently, and the mechanism is revealed as rebonding of unclosed dangling bonds at the ends of broken tubes [18.197]. Based on this interesting phenomenon, mechanochemical nanorobotic assembly has been performed [18.197].

Assembly of nanotubes is a fundamental technology for enabling nanodevices. The most important tasks include the connection of nanotubes and placing of nanotubes onto electrodes. Pure nanotube circuits [18.198] created by interconnecting nanotubes of different diameters and chirality could lead to further size reductions in devices. Nanotube intermolecular and intramolecular junctions are basic elements for such systems [18.199]. Room-temperature (RT) single-electron transistors (SETs) [18.200] have been shown with a short (≈ 20 nm) nanotube section that is created by inducing local barriers into the tube with an AFM, and Coulomb charging has been observed. With a cross-junction of two single-walled carbon nanotubes (SWNTs) (semiconducting/metallic), three- and four-terminal electronic devices have been made [18.201]. A suspended cross-junction can function as an electromechanical nonvolatile memory [18.202].

Although some kinds of junctions have been synthesized with chemical methods, there is no evidence yet showing that a self-assembly-based approach can provide more complex structures. SPMs were also used to fabricate junctions, but they are limited to a 2-D plane. We have presented 3-D nanorobotic manipulation-based nanoassembly, which is a promising strategy both for the fabrication of nanotube junctions and for the construction of more complex nanodevices with such junctions as well.

Nanotube junctions can be classified into different types by the kinds of components (SWNTs or MWNTs), geometric configurations (V kink, I, X cross, T, Y branch, and 3-D junctions) conductivity (metallic or semiconducting), and connection methods [intermolecular (connected with van der Waals force, EBID, etc.) or intramolecular (connected with chemical bonds) junctions].

tions]. Here we show the fabrication of several kinds of MWNT junctions by emphasizing the connection methods. These methods will also be effective for SWNT junctions. Figure 18.19 shows CNT junctions constructed by connecting with van der Waals forces (a), joining by electron-beam-induced deposition (b), and bonding through mechanochemistry (c).

Figure 18.19a shows a T-junction connected with van der Waals forces, which is fabricated by positioning the tip of an MWNT onto another MWNT until they form a bond. The contact is checked by measuring the shear connection force.

EBID provides a soldering method to obtain stronger nanotube junctions than those connected through van der Waals forces. Hence, if the strength of nanostructures is emphasized, EBID can be applied. Figure 18.19b shows an MWNT junction connected through EBID, in which the upper MWNT is a single one with 20 nm in diameter and the lower one is a bundle of MWNTs with an extruded single CNT with $\varnothing 30$ nm. The development of conventional EBID has been limited by the expensive electron filament used and low productivity. We have presented a parallel EBID system by using CNTs as emitters because of their excellent field-emission properties [18.203]. The feasibility of parallel EBID is presented. It is a promising strategy for large-scale fabrications of nanotube junctions. As in its macroscale counterpart, welding, EBID works by adding materials to obtain stronger connections, but in some cases, the added material might influence normal functions for nanosystems. So, EBID is mainly applied to nanostructures rather than nanomechanisms.

To construct stronger junctions without adding additional materials, mechanochemical nanorobotic assembly is an important strategy. Mechanochemical nanorobotic assembly is based on solid-phase chemical reactions, or mechanosynthesis, which is defined as chemical synthesis controlled by mechanical systems operating with atomic-scale precision, enabling direct positional selection of reaction sites [18.18]. By picking up atoms with dangling bonds rather than natural atoms only, it is easier to form primary bonds, which provides a simple but strong connection. Destructive fabrication provides a way to form dangling bonds at the ends of broken tubes. Some of the dangling bonds may close with neighboring atoms, but generally a few bonds will remain dangling. A nanotube with dangling bonds at its end will bind more easily to another to form intramolecular junctions. Figure 18.19c shows such a junction.

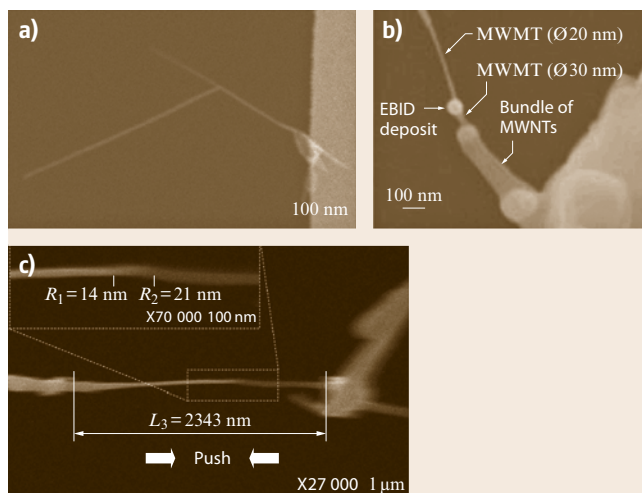


Fig. 18.19a–c MWNT junctions. (a) MWNTs connected with van der Waals forces. (b) MWNTs joined with EBID. (c) MWNTs bonded with a mechanochemical reaction

Three-dimensional nanorobotic manipulation has opened a new route for the structuring and assembly of nanotubes into nanodevices. However, at present nanomanipulation is still performed in a serial manner with master–slave control, which is not a large-scale production-oriented technique. Nevertheless, with advances in the exploration of mesoscopic physics, better control on the synthesis of nanotubes, more accurate actuators, and effective tools for manipulation, high-speed and automatic nanoassembly will be possible. Another approach might be parallel assembly by positioning building blocks with an array of probes [18.204] and joining them together simultaneously, e.g., with the parallel EBID [18.191] we presented. Further steps might progress towards exponential assembly [18.205], and in the far future to self-replicating assembly [18.18].

Nanorobotic Assembly of Nanocoils

The construction of nanocoil-based NEMS involves the assembly of as-grown or as-fabricated nanocoils, which is a significant challenge from a fabrication standpoint. Focusing on the unique aspects of manipulating nanocoils due to their helical geometry, high elasticity, single-end fixation, and strong adhesion of the coils to the substrate from wet etching, a series of new processes has been presented using a manipulator (MM3A, Kleindiek) installed in an SEM (Zeiss DSM962). As-fabricated SiGe/Si bilayer nanocoils (thickness 20 nm without Cr layer or 41 nm with Cr layer; diameter $D = 3.4 \mu\text{m}$) are manipulated. Special tools have been

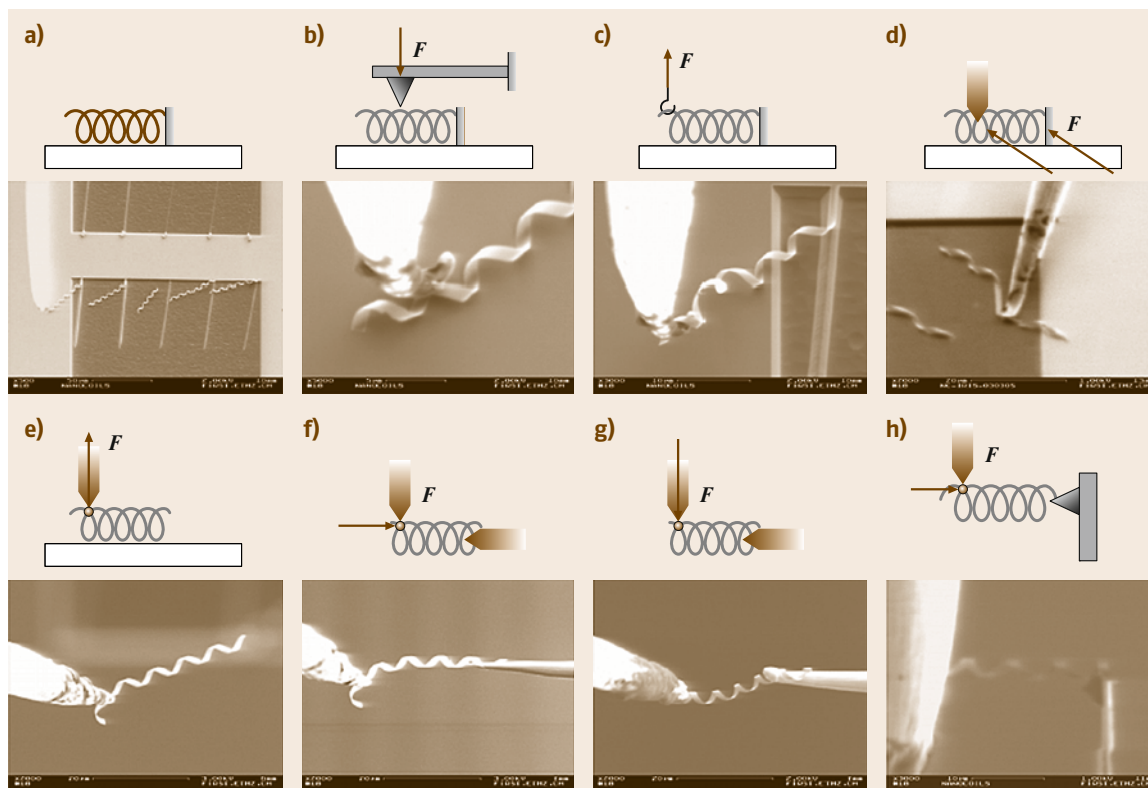


Fig. 18.20a–h Nanorobotic manipulation of nanocoils (a) original state, (b) compressing/releasing, (c) hooking, (d) lateral pushing/breaking, (e) picking, (f) placing/inserting, (g) bending, and (h) pushing and pulling

fabricated including a nanohook prepared by controlled *tip-crashing* of a commercially available tungsten sharp probe (Picoprobe T-4-10–1 mm and T-4-10) onto a substrate, and a *sticky* probe prepared by tip dipping into a double-sided SEM silver conductive tape (Ted Pella, Inc.). As shown in Fig. 18.20, experiments demonstrate that nanocoils can be released from a chip by lateral pushing, picked up with a nanohook or a *sticky* probe, and placed between the probe/hook and another probe or an AFM cantilever (Nano-probe, NP-S). Axial pulling/pushing, radial compressing/releasing, and bending/buckling have also been demonstrated. These processes have shown the effectiveness of manipulation for the characterization of coil-shaped nanostructures and their assembly for NEMS, which have been otherwise unavailable.

Configurations of nanodevices based on individual nanocoils are shown in Fig. 18.21. Cantilevered nanocoils as shown in Fig. 18.21a can serve as nanosprings. Nanoelectromagnets, chemical sensors, and nanoinductors involve nanocoils bridged between

two electrodes as shown in Fig. 18.21b. Electromechanical sensors can use a similar configuration but with one end connected to a moveable electrode as shown in Fig. 18.21c. Mechanical stiffness and electric conductivity are fundamental properties for these devices that must be further investigated.

As shown in Fig. 18.20h, axial pulling is used to measure the stiffness of a nanocoil. A series of SEM images are analyzed to extract the AFM tip displacement and the nanospring deformation, i. e., the relative displacement of the probe from the AFM tip. From this displacement data and the known stiffness of the AFM cantilever, the tensile force acting on the nanospring versus the nanospring deformation was plotted. The deformation of the nanospring was measured relative to the first measurement point. This was necessary because the proper attachment of the nanospring to the AFM cantilever must be verified. Afterwards, it was not possible to return to the point of zero deformation. Instead, the experimental data as presented in Fig. 18.21d has been shifted such that with the calculated linear elastic spring

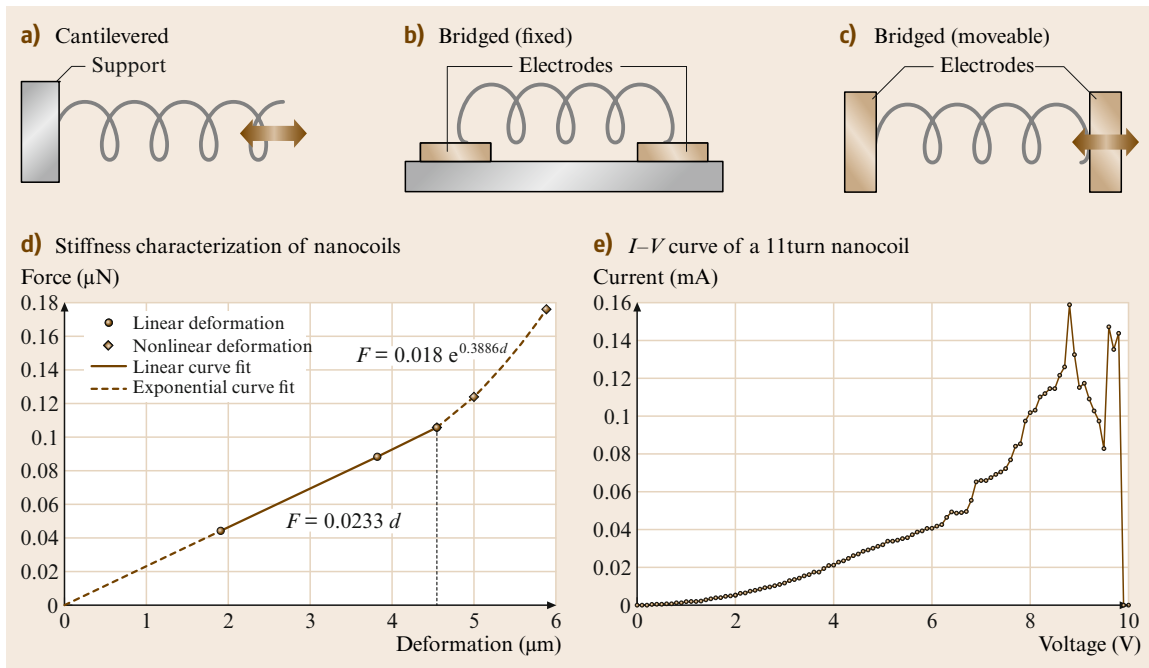


Fig. 18.21a–e Nanocoil-based devices. Cantilevered nanocoils (a) can serve as nanosprings. Nanoelectromagnets, chemical sensors, and nanoinductors involve nanocoils bridged between two electrodes (b). Electromechanical sensors can use a similar configuration but with one end connected to a moveable electrode (c). Mechanical stiffness (d) and electric conductivity (e) are basic properties of interest for these devices

stiffness the line begins at zero force and zero deformation. From Fig. 18.21d, the stiffness of the spring was estimated to be 0.0233 N/m. The linear elastic region of the nanospring extends to a deformation of 4.5 μm . An exponential approximation was fitted to the nonlinear region. When the applied force reached 0.176 μN , the attachment between the nanospring and the AFM cantilever broke. Finite-element simulation (ANSYS 9.0) was used to validate the experimental data [18.173]. Since the exact region of attachment cannot be identified from the SEM images, simulations were conducted for 4, 4.5, and 5 turns to estimate the possible range according to the apparent number of turns of the nanospring. The nanosprings in the simulations were fixed at one end and had an axial load of 0.106 μN applied at the other end. The simulation results for the spring with four turns yield a stiffness of 0.0302 N/m; for the nanospring with 5 turns it is 0.0191 N/m. The measured stiffness falls within this range with 22.0% above the minimum value and 22.8% below the maximum value, and very close to the stiffness of a 4.5-turn nanospring, which has a stiffness of 0.0230 N/m according to the simulation.

Figure 18.21e shows the results from electrical characterization experiments on a nanospring with 11 turns using the configuration shown in Fig. 18.20g. The I - V curve is nonlinear, which may be caused by the resistance change of the semiconductive bilayer due to ohmic heating. Another possible reason is the decrease in contact resistance caused by thermal stress. The maximum current was found to be 0.159 mA under an 8.8 V bias. Higher voltage causes the nanospring to *blow off*. From the fast scanning screen of the SEM, an extension of the nanospring on probes was observed around the peak current so that the current does not drop abruptly. At 9.4 V, the extended nanospring is broken down, causing an abrupt drop in the I - V curve.

From fabrication and characterization results, the helical nanostructures appear to be suitable to function as inductors. They would allow further miniaturization compared to state-of-the-art microinductors. For this purpose, higher doping of the bilayer and an additional metal layer would result in the required conductance. Conductance, inductance, and quality factor can be further improved if, after curling up, additional metal is electroplated onto the helical structures. Moreover,

a semiconductive helical structure, when functionalized with binding molecules, can be used for chemical sensing under the same principle as demonstrated with other types of nanostructures. With bilayers in the range of a few monolayers, the resulting structures would exhibit very high surface-to-volume ratio with the whole surface exposed to an incoming analyt.

18.8.4 Nanorobotic Devices

Nanorobotic devices involve tools, sensors, and actuators at the nanometer scale. Shrinking device size makes it possible to manipulate nanosized objects with nanosized tools, measure mass in femtogram ranges, sense force at piconewton scales, and induce GHz motion, among other amazing advancements.

Top-down and bottom-up strategies for manufacturing such nanodevices have been independently

investigated by a variety of researchers. Top-down strategies are based on nanofabrication and include technologies such as nanolithography, nanoimprinting, and chemical etching. Presently, these are 2-D fabrication processes with relatively low resolution. Bottom-up strategies are assembly-based techniques. At present, these strategies include such techniques as self-assembly, dip-pen lithography, and directed self-assembly. These techniques can generate regular nanopatterns at large scales. With the ability to position and orient nanoscale objects, nanorobotic manipulation is an enabling technology for structuring, characterizing and assembling many types of nanosystems [18.189]. By combining bottom-up and top-down processes, a hybrid nanorobotic approach based on nanorobotic manipulation provides a third way to fabricate NEMS by structuring as-grown nanomaterials or nanostructures. This new nanomanufacturing technique can be used

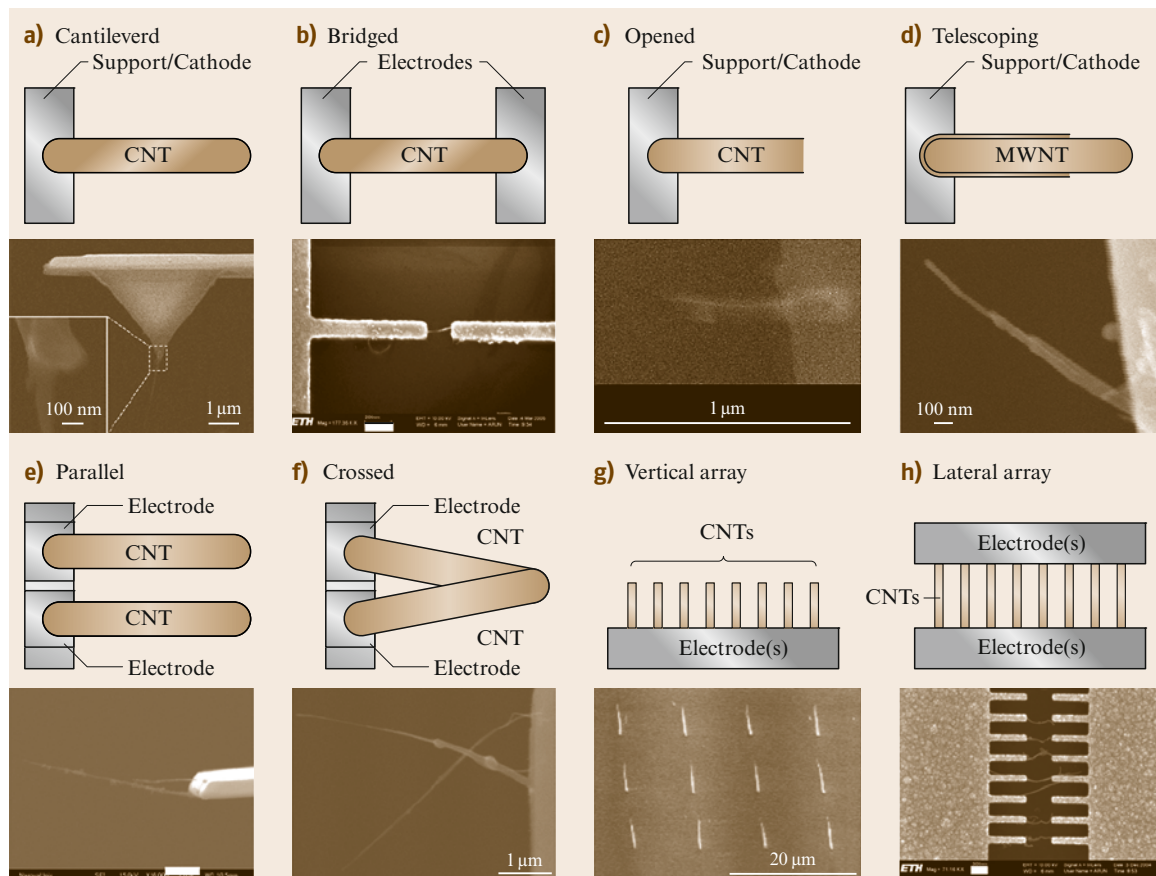


Fig. 18.22a–h Configurations of individual nanotube-based NEMS. Scale bars: (a) 1 μm (inset: 100 nm), (b) 200 nm, (c) 1 μm , (d) 100 nm, (e) and (f) 1 μm , (g) 20 μm , and (h) 300 nm. All examples are from the authors' work

to create complex 3-D nanodevices with such building blocks. Nanomaterial science, bio-nanotechnology, and nanoelectronics will also benefit from advances in nanorobotic assembly.

The configurations of nanotools, sensors, and actuators based on individual nanotubes that have been experimentally demonstrated are summarized as shown in Fig. 18.22.

For detecting deep and narrow features on a surface, cantilevered nanotubes (Fig. 18.22a, [18.191]) have been demonstrated as probe tips for an AFM [18.206], a STM, and other types of SPM. Nanotubes provide ultrasmall diameters, ultralarge aspect ratios, and excellent mechanical properties. Manual assembly, and direct growth [18.207] have proven effective for their construction. Cantilevered nanotubes have also been demonstrated as probes for the measurement of ultrasmall physical quantities, such as femtogram masses [18.162], piconewton-order force sensors, and mass flow sensors [18.189] on the basis of their static deflections or change of resonant frequencies detected within an electron microscope. Deflections cannot be measured from micrographs in real time, which limits the application of this kind of sensor. Interelectrode distance changes cause emission current variation of a nanotube emitter and may serve as a candidate to replace microscope images. Bridged individual nanotubes (Fig. 18.22b, [18.177]) have been the basis for electric characterization. Opened nanotubes (Fig. 18.22c, [18.208]) can serve as atomic or molecular containers, thermometers [18.209], or spot welders [18.210]. The electrostatic deflection of a nanotube has been used to construct a relay [18.211]. A new family of nanotube actuators can be constructed by taking advantage of the ultralow interlayer friction of

a multiwalled nanotubes. Linear bearings based on telescoping nanotubes and a microactuator with a nanotube as a rotation bearing have been demonstrated [18.195, 212], and batch fabrication has been realized based on dielectrophoretically assembled arrays [18.213]. A preliminary experiment on a promising nanotube linear motor with field-emission current serving as position feedback has been shown with nanorobotic manipulation (Fig. 18.22d, [18.208]). Cantilevered dual nanotubes have been demonstrated as nanotweezers [18.214] and nanoscissors (Fig. 18.22e) [18.179] by manual and nanorobotic assembly, respectively. Based on electric resistance change under different temperatures, nanotube thermal probes (Fig. 18.22f) have been demonstrated for measuring the temperature at precise locations. These thermal probes are more advantageous than nanotube-based thermometers because the latter require TEM imaging. The integration of the aforementioned devices can be realized using the configurations shown in Fig. 18.22g,h [18.177]. The arrays of individual nanotubes can also be used to fabricate nanosensors, such as position encoders [18.215].

Nanotube-based NEMS remains a rich research field with a large number of open problems. New materials such as nanowires, nanobelts, and polymer at the nanoscale will enable a new family of sensors and actuators for the detection and actuation of ultrasmall quantities or objects with ultrahigh precision and frequencies. Through random spreading, direct growth, optical tweezers, and nanorobotic manipulation, prototypes have been demonstrated. However, for integration into NEMS, self-assembly processes will become increasingly important. Among them, we believe that dielectrophoretic nanoassembly will play a significant role for large-scale production of regular 2-D structures.

18.9 Conclusions

Despite the claims of many *futurists*, such as Isaac Asimov's legendary submarine Proteus inside the human body [18.216] and Robert A. Freitas's nanomedical robots [18.217], the form that micro/nanorobots of the future will take and the tasks they will actually perform remain unclear. However, it is clear that technology is progressing towards the construction of intelligent sensors, actuators, and systems on small scales. These will serve as both the tools to be used for fabricating future micro/nanorobots as

well as the components from which these robots may be developed. Shrinking device size to these dimensions presents many fascinating opportunities such as manipulating nano-objects with nanotools, measuring mass in femtogram ranges, sensing forces at piconewton scales, and inducing GHz motion, among other new possibilities waiting to be discovered. These capabilities will, of course, drive the tasks that future micro/nanorobots constructed by and with MEMS/NEMS will perform.

References

- 18.1 R.P. Feynman: There's plenty of room at the bottom, *Caltech Eng. Sci.* **23**, 22–36 (1960)
- 18.2 R.S. Muller: Microdynamics, *Sens. Actuat. A* **21–23**, 1–8 (1990)
- 18.3 A.M. Flynn, R.A. Brooks, W.M. Wells, III, D.S. Barrett: The world's largest one cubic inch robot, *Proc. of IEEE 2nd Int. Workshop on Micro Electro Mechanical Systems (IEEE, Piscataway 1989)* pp. 98–101
- 18.4 W. Trimmer, R. Jebens: Actuators for micro robots, *Proc. of the 1989 IEEE Int. Conf. on Robotics and Automation (IEEE, Piscataway 1989)* pp. 1547–1552
- 18.5 S. Fatikow, U. Rembold: An automated microrobot-based desktop station for micro assembly and handling of micro-objects, *IEEE Conf. on Emerging Technologies and Factory Automation (EFTA'96) (IEEE, Piscataway 1996)* pp. 586–592
- 18.6 B.J. Nelson, Y. Zhou, B. Vikramaditya: Sensor-based microassembly of hybrid MEMS devices, *IEEE Contr. Syst. Mag.* **18**, 35–45 (1998)
- 18.7 K. Suzumori, T. Miyagawa, M. Kimura, Y. Hasegawa: Micro inspection robot for 1-in pipes, *IEEE/ASME Trans. Mechatron.* **4**, 286–292 (1999)
- 18.8 M. Takeda: Applications of MEMS to industrial inspection, *Proc. 14th IEEE Int. Conf. on Micro Electro Mechanical Systems (IEEE, Piscataway 2001)* pp. 182–191
- 18.9 T. Frank: Two-Axis electrodynamic micropositioning devices, *J. Micromech. Microeng.* **8**, 114–118 (1989)
- 18.10 N. Kawahara, N. Kawahara, T. Suto, T. Hirano, Y. Ishikawa, T. Kitahara, N. Ooyama, T. Ataka: Microfactories: New applications of micromachine technology to the manufacture of small products, *Res. J. Microsyst. Technol.* **3**, 37–41 (1997)
- 18.11 Y. Sun, B.J. Nelson: Microrobotic cell injection, *Proc. of the 2001 IEEE International Conf. on Robotics and Automation (ICRA2001) (IEEE, Piscataway 2001)* pp. 620–625
- 18.12 P. Dario, M.C. Carrozza, L. Lencioni, B. Magnani, S. Dapos Attanasio: A micro robotic system for colonoscopy, *Proc. 1997 Int. Conf. on Robotics and Automation (IEEE, Piscataway 1997)* pp. 1567–1572
- 18.13 F. Tendick, S.S. Sastry, R.S. Fearing, M. Cohn: Application of micromechatronics in minimally invasive surgery, *IEEE/ASME Trans. Mechatron.* **3**, 34–42 (1998)
- 18.14 G. Iddan, G. Meron, A. Glukhovsky, P. Swain: Wire-less capsule endoscopy, *Nature* **405**, 417 (2000)
- 18.15 K.B. Yesin, K. Vollmers, B.J. Nelson: Analysis and design of wireless magnetically guided microrobots in body fluids, *Proc. 2004 IEEE Int. Conf. on Robotics and Automation (IEEE, Piscataway 2004)* pp. 1333–1338
- 18.16 M.C. Roco, R.S. Williams, P. Alivisatos: *Nanotechnology Research Directions: Interagency Working Group on Nanoscience, Engineering and Technology (IWGN) (Workshop Report) (Kluwer, Dordrecht 2000)*
- 18.17 M.L. Downey, D.T. Moore, G.R. Bachula, D.M. Etter, E.F. Carey, L.A. Perine: National Nanotechnology Initiative: Leading to the Next Industrial Revolution, A Report by the Interagency Working Group on Nanoscience, Engineering and Technology (Committee on Technology, National Science and Technology Council, Washington 2000)
- 18.18 K. Drexler: *Nanosystems: Molecular Machinery, Manufacturing and Computation* (Wiley, New York 1992)
- 18.19 G. Binnig, H. Rohrer, C. Gerber, E. Weibel: Surface studies by scanning tunneling microscopy, *Phys. Rev. Lett.* **49**, 57–61 (1982)
- 18.20 W.F. Degrado: Design of peptides and proteins, *Adv. Protein Chem.* **39**, 51–124 (1998)
- 18.21 G.M. Whitesides, B. Grzybowski: Self-assembly at all scales, *Science* **295**, 2418–2421 (2002)
- 18.22 R. Fearing: Survey of sticking effects for micro-parts, *Proc. 1995 IEEE/RSJ Int. Conf. Int. Robots and Systems (IEEE, Piscataway 1995)* pp. 212–217
- 18.23 E.L. Wolf: *Nanophysics and Nanotechnology* (WILEY-VCH, Weinheim 2004)
- 18.24 C.-J. Kim, A.P. Pisano, R.S. Muller: Silicon-processed overhanging microgripper, *IEEE/ASME J. MEMS* **1**, 31–36 (1992)
- 18.25 C. Liu, T. Tsao, Y.-C. Tai, C.-M. Ho: Surface micromachined magnetic actuators, *Proc. 7th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 1994)* pp. 57–62
- 18.26 J. Judy, D.L. Polla, W.P. Robbins: A linear piezoelectric stepper motor with submicron displacement and centimeter travel, *IEEE Trans. Ultrason. Ferroelectr. Freq. Contr.* **37**, 428–437 (1990)
- 18.27 K. Nakamura, H. Ogura, S. Maeda, U. Sangawa, S. Aoki, T. Sato: Evaluation of the micro wobbler motor fabricated by concentric build-up process, *Proc. 8th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 1995)* pp. 374–379
- 18.28 A. Teshigahara, M. Watanabe, N. Kawahara, I. Ohtsuka, T. Hattori: Performance of a 7-mm microfabricated car, *IEEE/ASME J. MEMS* **4**, 76–80 (1995)
- 18.29 K.R. Udayakumar, S.F. Bart, A.M. Flynn, J. Chen, L.S. Tavrow, L.E. Cross, R.A. Brooks, D.J. Ehrlich: Ferroelectric thin film ultrasonic micromotors, *Proc. 4th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 1991)* pp. 109–113
- 18.30 T. Ebefors, G. Stemme: Microrobotics. In: *The MEMS Handbook*, ed. by M. Gad-el-Hak (CRC, Boca Raton 2002)
- 18.31 P. Dario, R. Valleggi, M.C. Carrozza, M.C. Montesi, M. Cocco: Review – Microactuators for microrobots:

- A critical survey, *J. Micromech. Microeng.* **2**, 141–157 (1992)
- 18.32 I. Shimoyama: Scaling in microrobots, *Proc. IEEE/RSJ Intelligent Robots and Systems* (IEEE, Piscataway 1995) pp. 208–211
- 18.33 R.S. Fearing: Powering 3-dimensional microrobots: power density limitations, tutorial on “Micro Mechatronics and Micro Robotics”, *Proc. 1998 IEEE Int. Conf. on Robotics and Automation* (IEEE, Piscataway 1998)
- 18.34 R.G. Gilbertson, J.D. Busch: A survey of micro-actuator technologies for future spacecraft missions, *J. Br. Interplanet. Soc.* **49**, 129–138 (1996)
- 18.35 M. Mehregany, P. Nagarkar, S.D. Senturia, J.H. Lang: Operation of microfabricated harmonic and ordinary side-drive motors, *Proc. 3rd IEEE Int. Conf. Micro Electro Mechanical Systems* (IEEE, Piscataway 1990) pp. 1–8
- 18.36 Y.C. Tai, L.S. Fan, R.S. Muller: IC-processed micro-motors: design, technology, and testing, *Proc. 2nd IEEE Int. Conf. Micro Electro Mechanical Systems* (IEEE, Piscataway 1989) pp. 1–6
- 18.37 T. Ohnstein, T. Fukiura, J. Ridley, U. Bonne: Micromachined silicon microvalve, *Proc. 3rd IEEE Int. Conf. Micro Electro Mechanical Systems* (IEEE, Piscataway 1990) pp. 95–99
- 18.38 L.Y. Chen, S.L. Zhang, J.J. Yao, D.C. Thomas, N.C. MacDonald: Selective chemical vapor deposition of tungsten for microdynamic structures, *Proc. 2nd IEEE Int. Conf. Micro Electro Mechanical Systems* (IEEE, Piscataway 1989) pp. 82–87
- 18.39 K. Yanagisawa, H. Kuwano, A. Tago: An electromagnetically driven microvalve, *Proc. 7th Int. Conf. on Solid-State Sensors and Actuators* (IEEE, Piscataway 1993) pp. 102–105
- 18.40 M. Esashi, S. Shoji, A. Nakano: Normally close microvalve and micropump fabricated on a silicon wafer, *Proc. 2nd IEEE Int. Conf. Micro Electro Mechanical Systems* (IEEE, Piscataway 1989) pp. 29–34
- 18.41 R. Petrucci, K. Simmons: An introduction to piezoelectric crystals. In: *Sensors Magazine* (Helmert, Peterborough 1994) pp. 26–
- 18.42 J. Goldstein, D. Newbury, D. Joy, C. Lyman, P. Echlin, E. Lifshin, L. Sawyer, J. Michael: *Scanning Electron Microscopy and X-ray Microanalysis* (Kluwer Academic/Plenum, New York 2003)
- 18.43 G. Binnig, H. Rohrer: In touch with atoms, *Rev. Mod. Phys.* **71**, S324–S330 (1999)
- 18.44 G. Binnig, C.F. Quate, C. Gerber: Atomic force microscope, *Phys. Rev. Lett.* **56**, 93–96 (1986)
- 18.45 M.J. Doktycz, C.J. Sullivan, P.R. Hoyt, D.A. Pelletier, S. Wu, D.P. Allison: AFM imaging of bacteria in liquid media immobilized on gelatin coated mica surfaces, *Ultramicroscopy* **97**, 209–216 (2003)
- 18.46 S.A. Campbell: *The Science and Engineering of Microelectronic Fabrication* (Oxford Univ. Press, New York 2001)
- 18.47 C.J. Jaeger: *Introduction to Microelectronic Fabrication* (Prentice Hall, Upper Saddle River 2002)
- 18.48 J.D. Plummer, M.D. Deal, P.B. Griffin: *Silicon VLSI Technology* (Prentice Hall, Upper Saddle River 2000)
- 18.49 M. Gad-el-Hak (Ed.): *The MEMS Handbook* (CRC, Boca Raton 2002)
- 18.50 T.-R. Hsu: *MEMS and Microsystems Design and Manufacture* (McGraw-Hill, New York 2002)
- 18.51 G.T.A. Kovacs: *Micromachined Transducers Sourcebook* (McGraw-Hill, New York 1998)
- 18.52 G.T.A. Kovacs, N.I. Maluf, K.A. Petersen: Bulk micromachining of silicon, *Proc. IEEE Int. Conf. Robot. Autom.* **86**, 1536–1551 (1998)
- 18.53 P. Rai-Choudhury (Ed.): *Handbook of Microlithography, Micromachining and Microfabrication* (SPIE, Bellingham 1997)
- 18.54 S.Y. Chou: Nano-imprint lithography and lithographically induced self-assembly, *MRS Bull.* **26**, 512–517 (2001)
- 18.55 M.A. Herman: *Molecular Beam Epitaxy: Fundamentals and Current Status* (Springer, New York 1996)
- 18.56 J.S. Frood, G.J. Davis, W.T. Tsang: *Chemical Beam Epitaxy and Related Techniques* (Wiley, New York 1997)
- 18.57 S. Mahajan, K.S.S. Harsha: *Principles of Growth and Processing of Semiconductors* (McGraw-Hill, New York 1999)
- 18.58 C.A. Mirkin: Dip-pen nanolithography: automated fabrication of custom multicomponent, sub-100 nanometer surface architectures, *MRS Bull.* **26**, 535–538 (2001)
- 18.59 C.A. Harper: *Electronic Packaging and Interconnection Handbook* (McGraw-Hill, New York 2000)
- 18.60 K.F. Bohringer, R.S. Fearing, K.Y. Goldberg: Microassembly. In: *Handbook of Industrial Robotics*, ed. by S. Nof (Wiley, New York 1999) pp. 1045–1066
- 18.61 G. Yang, J.A. Gaines, B.J. Nelson: A supervisory wafer-level 3D microassembly system for hybrid MEMS fabrication, *J. Intell. Robot. Syst.* **37**, 43–68 (2003)
- 18.62 P. Dario, M. Carrozza, N. Croce, M. Montesi, M. Cocco: Non-traditional technologies for microfabrication, *J. Micromech. Microeng.* **5**, 64–71 (1995)
- 18.63 W. Benecke: Silicon microactuators: activation mechanisms and scaling problems, *Proc. IEEE Int. Conf. Solid-State Sensors and Actuators* (IEEE, Piscataway 1991) pp. 46–50
- 18.64 A. Menciassi, A. Eisinger, M. Mazzoni, P. Dario: A sensorized electro discharge machined superelastic alloy microgripper for micromanipulation: simulation and characterization, *Proc. 2002 IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (IEEE, Piscataway 2002) pp. 1591–1595
- 18.65 T.R. Hsu: Packaging design of microsystems and meso-scale devices, *IEEE Trans. Adv. Packag.* **23**, 596–601 (2000)

- 18.66 L. Lin: MEMS post-packaging by localized heating and bonding, *IEEE Trans. Adv. Packag.* **23**, 608–616 (2000)
- 18.67 A. Tixier, Y. Mita, S. Oshima, J.P. Gouy, H. Fujita: 3-D microsystem packaging for interconnecting electrical, optical and mechanical microdevices to the external world, *Proc. 13th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 2000)* pp. 698–703
- 18.68 M.J. Madou: *Fundamentals of Microfabrication* (CRC, Boca Raton 2002)
- 18.69 I. Shimoyama, O. Kano, H. Miura: 3D microstructures folded by Lorentz force, *Proc. 11th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 1998)* pp. 24–28
- 18.70 K.F. Bohringer, B.R. Donald, L. Kavraki, F.L. Lami-raux: Part orientation with one or two stable equilibria using programmable vector fields, *IEEE Trans. Robot. Autom.* **16**, 157–170 (2000)
- 18.71 V. Kaajakari, A. Lal: An electrostatic batch assembly of surface MEMS using ultrasonic triboelectricity, *Proc. 14th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 2001)* pp. 10–13
- 18.72 G. Yang, B.J. Nelson: Micromanipulation contact transition control by selective focusing and micro-force control, *Proc. 2003 IEEE Int. Conf. on Robotics and Automation (IEEE, Piscataway 2003)* pp. 3200–3206
- 18.73 G. Morel, E. Malis, S. Boudet: Impedance based combination of visual and force control, *Proc. 1998 IEEE Int. Conf. on Robotics and Automation (IEEE, Piscataway 1998)* pp. 1743–1748
- 18.74 F. Arai, D. Andou, T. Fukuda: Adhesion forces reduction for micro manipulation based on micro physics, *Proc. 9th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 1996)* pp. 354–359
- 18.75 Y. Zhou, B.J. Nelson: The effect of material properties and gripping force on micrograsping, *Proc. 2000 IEEE Int. Conf. Robotics and Automation (IEEE, Piscataway 2000)* pp. 1115–1120
- 18.76 K. Kurata: Mass production techniques for optical modules, *Proc. 48th IEEE Electronic Components and Technology Conf. (IEEE, Piscataway 1998)* pp. 572–580
- 18.77 V.T. Portman, B.-Z. Sandler, E. Zahavi: Rigid 6 × 6 parallel platform for precision 3-D micromanipulation: theory and design application, *IEEE Trans. Robot. Autom.* **16**, 629–643 (2000)
- 18.78 R.M. Haralick, L.G. Shapiro: *Computer and Robot Vision* (Addison-Wesley, Reading 1993)
- 18.79 A. Khotanzad, H. Banerjee, M.D. Srinath: A vision system for inspection of ball bonds and 2-D profile of bonding wires in integrated circuits, *IEEE Trans. Semicond. Manuf.* **7**, 413–422 (1994)
- 18.80 J.T. Feddema, R.W. Simon: CAD-driven microassembly and visual servoing, *Proc. 1998 IEEE Int. Conf. Robotics and Automation (IEEE, Piscataway 1998)* pp. 1212–1219
- 18.81 E. Trucco, A. Verri: *Introductory Techniques for 3-D Computer Vision* (Prentice Hall, Upper Saddle River 1998)
- 18.82 S. Hutchinson, G.D. Hager, P.I. Corke: A tutorial on visual servo control, *IEEE Trans. Robot. Autom.* **12**, 651–670 (1996)
- 18.83 B. Siciliano, L. Villani: *Robot Force Control* (Kluwer, Dordrecht 2000)
- 18.84 T. Yoshikawa: Force control of robot manipulators, *Proc. 2000 IEEE Int. Conf. Robotics and Automation (IEEE, Piscataway 2000)* pp. 220–226
- 18.85 J.A. Thompson, R.S. Fearing: Automating microassembly with ortho-tweezers and force sensing, *Proc. 2001 IEEE/RSJ Int. Conf. Int. Robots and Systems (IEEE, Piscataway 2001)* pp. 1327–1334
- 18.86 B.J. Nelson, P.K. Khosla: Force and vision resolvability for assimilating disparate sensory feedback, *IEEE Trans. Robot. Autom.* **12**, 714–731 (1996)
- 18.87 Y. Haddab, N. Chaillet, A. Bourjault: A microgripper using smart piezoelectric actuators, *Proc. 2000 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IEEE, Piscataway 2000)* pp. 659–664
- 18.88 D. Popa, B.H. Kang, J. Sin, J. Zou: Reconfigurable micro-assembly system for photonics applications, *Proc. 2002 IEEE Int. Conf. Robotics and Automation (IEEE, Piscataway 2002)* pp. 1495–1500
- 18.89 A.P. Lee, D.R. Ciarlo, P.A. Krulvitch, S. Lehew, J. Trevin, M.A. Northrup: A practical microgripper by fine alignment, eutectic bonding and SMA actuation, *Transducers'95 (IEEE, Piscataway 1995)* pp. 368–371
- 18.90 H. Seki: Modeling and impedance control of a piezoelectric bimorph microgripper, *Proc. 1992 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IEEE, Piscataway 1992)* pp. 958–965
- 18.91 W. Nogimori, K. Irisa, M. Ando, Y. Naruse: A laser-powered micro-gripper, *Proc. 10th IEEE Int. Conf. Micro Electro Mechanical Systems (IEEE, Piscataway 1997)* pp. 267–271
- 18.92 S. Fatikow, U. Rembold: *Microsystem Technology and Microrobotics* (Springer, Berlin, Heidelberg 1997)
- 18.93 T. Hayashi: Micro mechanism, *J. Robot. Mechatr.* **3**, 2–7 (1991)
- 18.94 S. Johansson: Micromanipulation for micro- and nanomanufacturing, *INRIA/IEEE Symp. on Emerging Technologies and Factory Automation (ETFA'95), Paris (1995)* pp. 3–8
- 18.95 K.-T. Park, M. Esashi: A multilink active catheter with polyimide-based integrated CMOS interface circuits, *J. MEMS* **8**, 349–357 (1999)
- 18.96 Y. Haga, Y. Tanahashi, M. Esashi: Small diameter active catheter using shape memory alloy, *Proc. IEEE 11th Int. Workshop on Micro Electro Mechanical Systems, Heidelberg (1998)* pp. 419–424

- 18.97 E.W.H. Jager, O. Inganas, I. Lundstrom: Microrobots for micrometer-size objects in aqueous media: Potential tools for single cell manipulation, *Science* **288**, 2335–2338 (2000)
- 18.98 E.W.H. Jager, E. Smela, O. Inganas: Microfabricating conjugated polymer actuators, *Science* **290**, 1540–1545 (2000)
- 18.99 J.W. Suh, S.F. Glander, R.B. Darling, C.W. Storment, G.T.A. Kovacs: Organic thermal and electrostatic ciliary microactuator array for object manipulation, *Sens. Actuat. A* **58**, 51–60 (1997)
- 18.100 E. Smela, M. Kallenbach, J. Holdenried: Electrochemically driven polypyrrole bilayers for moving and positioning bulk micromachined silicon plates, *J. MEMS* **8**, 373–383 (1999)
- 18.101 S. Konishi, H. Fujita: A conveyance system using air flow based on the concept of distributed micro motion systems, *IEEE J. MEMS* **3**, 54–58 (1994)
- 18.102 M. Ataka, A. Omodaka, N. Takeshima, H. Fujita: Fabrication and operation of polyimide bimorph actuators for a ciliary motion system, *J. MEMS* **2**, 146–150 (1993)
- 18.103 G.-X. Zhou: Swallowable or implantable body temperature telemeter-body temperature radio pill, *Proc. IEEE 15th Annual Northeast Bioengineering Conf.* (1989) pp. 165–166
- 18.104 A. Uchiyama: Endoradiosonde Needs Micro Machine Technology, *Proc. IEEE 6th Int. Symp. on Micro Machine and Human Science (MHS '95)* (IEEE, Nagoya 1995) pp. 31–37
- 18.105 Y. Carts-Powell: Tiny Camera in a Pill Extends Limits of Endoscopy, *SPIE: OE-Reports* Aug. (2000) (available on Internet at: <http://www.spie.org>)
- 18.106 R. Yeh, E.J.J. Kruglick, K.S.J. Pister: Surface-micromachined components for articulated microrobots, *J. MEMS* **5**, 10–17 (1996)
- 18.107 P.E. Kladitis, V.M. Bright, K.F. Harsh, Y.C. Lee: Prototype Microrobots for micro positioning in a manufacturing process and micro unmanned vehicles, *Proc. IEEE 12th Int. Conf. on Micro Electro Mechanical Systems (MEMS'99)*, Orlando (1999) pp. 570–575
- 18.108 D. Ruffieux, N.F. d. Rooij: A 3-DoF bimorph actuator array capable of locomotion, *13th European Conf. on Solid-State Transducers (Euroensors XIII)*, Hague (1999) pp. 725–728
- 18.109 J.-M. Breguet, P. Renaud: A 4 degrees-of-freedom microrobot with nanometer resolution, *Robotics* **14**, 199–203 (1996)
- 18.110 A. Flynn, L.S. Tavrow, S.F. Bart, R.A. Brooks, D.J. Ehrlich, K.R. Udayakumar, L.E. Cross: Piezo-electric micromotors for microrobots, *IEEE J. MEMS* **1**, 44–51 (1992)
- 18.111 A. Teshigahara, M. Watanabe, N. Kawahara, Y. Ohtsuka, T. Hattori: Performance of a 7mm microfabricated car, *IEEE/ASME J. MEMS* **4**, 76–80 (1995)
- 18.112 T. Ebefors, J. Mattson, E. Kalvesten, G. Stemme: A walking silicon micro-robot, *10th Int. Conf. on Solid-State Sensors and Actuators (Transducers'99)*, Sendai (1999) pp. 1202–1205
- 18.113 N. Miki, I. Shimoyama: Flight performance of micro-wings rotating in an alternating magnetic field, *Proc. IEEE 12th Int. Conf. on Micro Electro Mechanical Systems (MEMS'99)*, Orlando (1999) pp. 153–158
- 18.114 Mainz: Micro-motors: The World's Tiniest Helicopter, in <http://www.imm-mainz.de/english/developm/products/hubi.html> Orlando (1999)
- 18.115 K.I. Arai, W. Sugawara, T. Honda: *Magnetic small flying machines*, Tech. Digest Transducers'95 and Euroensors IX, Stockholm (1995) pp. 316–319
- 18.116 T. Fukuda, A. Kawamoto, F. Arai, H. Matsuura: Mechanism and swimming experiment of micro mobile robot in water, *Proc. IEEE 7th Int. Workshop on Micro Electro Mechanical Systems (MEMS'94)*, Oiso (1994) pp. 273–278
- 18.117 I. Shimoyama: Hybrid system of mechanical parts and living organisms for microrobots, *Proc. IEEE 6th Int. Symp. on Micro Machine and Human Science (MHS '95)* (IEEE, Nagoya 1995) p. 55
- 18.118 A. Ashkin: Acceleration and trapping of particles by radiation pressure, *Phys. Rev. Lett.* **24**, 156–159 (1970)
- 18.119 T.N. Bruican, M.J. Smyth, H.A. Crissman, G.C. Salzman, C.C. Stewart, J.C. Martin: Automated single-cell manipulation and sorting by light trapping, *Appl. Opt.* **26**, 5311–5316 (1987)
- 18.120 J. Conia, B.S. Edwards, S. Voelkel: The micro-robotic laboratory: Optical trapping and scissoring for the biologist, *J. Clin. Lab. Anal.* **11**, 28–38 (1997)
- 18.121 W.H. Wright, G.J. Sonek, Y. Tadir, M.W. Berns: Laser trapping in cell biology, *IEEE J. Quant. Electron.* **26**, 2148–2157 (1990)
- 18.122 F. Arai, K. Morishima, T. Kasugai, T. Fukuda: Bio-micromanipulation (new direction for operation improvement), *Proc. of 1997 IEEE/RSJ Int. Conf. on Intelligent Robotics and Systems* (IEEE, Piscataway 1997) pp. 1300–1305
- 18.123 M. Nishioka, S. Katsura, K. Hirano, A. Mizuno: Evaluation of cell characteristics by step-wise orientational rotation using optoelectrostatic micromanipulation, *IEEE Trans. Ind. Appl.* **33**, 1381–1388 (1997)
- 18.124 M. Washizu, Y. Kurahashi, H. Iochi, O. Kurosawa, S. Aizawa, S. Kudo, Y. Magariyama, H. Hotani: Dielectrophoretic measurement of bacterial motor characteristics, *IEEE Trans. Ind. Appl.* **29**, 286–294 (1993)
- 18.125 Y. Kimura, R. Yanagimachi: Intracytoplasmic sperm injection in the mouse, *Biol. Reprod.* **52**, 709–720 (1995)
- 18.126 M. Mischel, A. Voss, H.A. Pohl: Cellular spin resonance in rotating electric fields, *J. Biol. Phys.* **10**, 223–226 (1982)

- 18.127 W.M. Arnold, U. Zimmermann: Electro-Rotation: Development of a technique for dielectric measurements on individual cells and particles, *J. Electrostat.* **21**, 151–191 (1988)
- 18.128 Y. Sun, B.J. Nelson: Autonomous injection of biological cells using visual servoing, *Int. Symp. on Experimental Robotics (ISER 2000)*, *Lect. Notes Contr. Inform. Sci.* (2000) pp. 175–184
- 18.129 Y. Sun, B.J. Nelson, D.P. Potasek, E. Enikov: A bulk microfabricated multi-axis capacitive cellular force sensor using transverse comb drives, *J. Micromech. Microeng.* **12**, 832–840 (2002)
- 18.130 Y. Sun, K. Wan, K.P. Roberts, J.C. Bischof, B.J. Nelson: Mechanical property characterization of mouse zona pellucida, *IEEE Trans. Nanobiosci.* **2**, 279–286 (2003)
- 18.131 A. Ashkin, J.M. Dziedzic: Optical trapping and manipulation of viruses and bacteria, *Science* **235**, 1517–1520 (1987)
- 18.132 F.H.C. Crick, A.F.W. Hughes: The physical properties of cytoplasm: A study by means of the magnetic particle method, Part I. *Experimental Exp. Cell Res.* **1**, 37–80 (1950)
- 18.133 M.F. Yu, M.J. Dyer, G.D. Skidmore, H.W. Rohrs, X.K. Lu, K.D. Ausman, J.R.V. Ehr, R.S. Ruoff: Three-dimensional manipulation of carbon nanotubes under a scanning electron microscope, *Nanotechnology* **10**, 244–252 (1999)
- 18.134 L.X. Dong, F. Arai, T. Fukuda: 3D nanorobotic manipulation of nano-order objects inside SEM, *Proc. of 2000 Int. Symp. on Micromechatronics and Human Science (MHS2000)* (IEEE, Piscataway 2000) pp. 151–156
- 18.135 D.M. Eigler, E.K. Schweizer: Positioning single atoms with a scanning tunneling microscope, *Nature* **344**, 524–526 (1990)
- 18.136 P. Avouris: Manipulation of matter at the atomic and molecular levels, *Acc. Chem. Res.* **28**, 95–102 (1995)
- 18.137 M.F. Crommie, C.P. Lutz, D.M. Eigler: Confinement of electrons to quantum corrals on a metal surface, *Science* **262**, 218–220 (1993)
- 18.138 L.J. Whitman, J.A. Stroscio, R.A. Dragoset, R.J. Celotta: Manipulation of adsorbed atoms and creation of new structures on room-temperature surfaces with a scanning tunneling microscope, *Science* **251**, 1206–1210 (1991)
- 18.139 I.-W. Lyo, P. Avouris: Field-induced nanometer-scale to atomic-scale manipulation of silicon surfaces with the STM, *Science* **253**, 173–176 (1991)
- 18.140 G. Dujardin, R.E. Walkup, P. Avouris: Dissociation of individual molecules with electrons from the tip of a scanning tunneling microscope, *Science* **255**, 1232–1235 (1992)
- 18.141 T.-C. Shen, C. Wang, G.C. Abeln, J.R. Tucker, J.W. Lyding, P. Avouris, R.E. Walkup: Atomic-scale desorption through electronic and vibrational-excitation mechanisms, *Science* **268**, 1590–1592 (1995)
- 18.142 M.T. Cuberes, R.R. Schittler, J.K. Gimzewski: Room-temperature repositioning of individual C60 molecules at Cu steps: operation of a molecular counting device, *Appl. Phys. Lett.* **69**, 3016–3018 (1996)
- 18.143 H.J. Lee, W. Ho: Single-bond formation and characterization with a scanning tunneling microscope, *Science* **286**, 1719–1722 (1999)
- 18.144 T. Yamamoto, O. Kurosawa, H. Kabata, N. Shimamoto, M. Washizu: Molecular surgery of DNA based on electrostatic micromanipulation, *IEEE Trans. IA* **36**, 1010–1017 (2000)
- 18.145 C. Haber, D. Wirtz: Magnetic tweezers for DNA micromanipulation, *Rev. Sci. Instrum.* **71**, 4561–4570 (2000)
- 18.146 D.M. Schaefer, R. Reifengerger, A. Patil, R.P. Andres: Fabrication of two-dimensional arrays of nanometer-size clusters with the atomic force microscope, *Appl. Phys. Lett.* **66**, 1012–1014 (1995)
- 18.147 T. Junno, K. Deppert, L. Montelius, L. Samuelson: Controlled manipulation of nanoparticles with an atomic force microscope, *Appl. Phys. Lett.* **66**, 3627–3629 (1995)
- 18.148 P.E. Sheehan, C.M. Lieber: Nanomachining, manipulation and fabrication by force microscopy, *Nanotechnology* **7**, 236–240 (1996)
- 18.149 C. Baur, B.C. Gazez, B. Koel, T.R. Ramachandran, A.A.G. Requicha, L. Zini: Robotic nanomanipulation with a scanning probe microscope in a networked computing environment, *J. Vac. Sci. Tech. B* **15**, 1577–1580 (1997)
- 18.150 A.A.G. Requicha: Nanorobots, NEMS, and nanoassembly, *Proc. IEEE* **91**, 1922–1933 (2003)
- 18.151 R. Resch, C. Baur, A. Bugacov, B.E. Koel, A. Madhukar, A.A.G. Requicha, P. Will: Building and manipulating 3-D and linked 2-D structures of nanoparticles using scanning force microscopy, *Langmuir* **14**, 6613–6616 (1998)
- 18.152 J. Hu, Z.-H. Zhang, Z.-Q. Ouyang, S.-F. Chen, M.-Q. Li, F.-J. Yang: Stretch and align virus in nanometer scale on an atomically flat surface, *J. Vac. Sci. Tech. B* **16**, 2841–2843 (1998)
- 18.153 M. Sitti, S. Horiguchi, H. Hashimoto: Controlled pushing of nanoparticles: modeling and experiments, *IEEE/ASME Trans. Mechatron.* **5**, 199–211 (2000)
- 18.154 M. Guthold, M.R. Falvo, W.G. Matthews, S. Paulson, S. Washburn, D.A. Erie, R. Superfine, J.F.P. Brooks, I.R.M. Taylor: Controlled manipulation of molecular samples with the nanoManipulator, *IEEE/ASME Trans. Mechatron.* **5**, 189–198 (2000)
- 18.155 G.Y. Li, N. Xi, M.M. Yu, W.K. Fung: Development of augmented reality system for AFM-based nanomanipulation, *IEEE/ASME Trans. Mechatron.* **9**, 358–365 (2004)

- 18.156 F. Arai, D. Andou, T. Fukuda: Micro manipulation based on micro physics—strategy based on attractive force reduction and stress measurement, Proc. of IEEE/RSJ Int. Conf. on Intelligent Robotics and Systems (IEEE, Piscataway 1995) pp. 236–241
- 18.157 H.W.P. Koops, J. Kretz, M. Rudolph, M. Weber, G. Dahm, K.L. Lee: Characterization and application of materials grown by electron-beam-induced deposition, Jpn. J. Appl. Phys. **33**, 7099–7107 (1994), Part 1
- 18.158 S. Iijima: Helical microtubules of graphitic carbon, Nature **354**, 56–58 (1991)
- 18.159 S.J. Tans, A.R.M. Verchueren, C. Dekker: Room-temperature transistor based on a single carbon nanotube, Nature **393**, 49–52 (1998)
- 18.160 R.H. Baughman, A.A. Zakhidov, W.A. de Heer: Carbon nanotubes—the route toward applications, Science **297**, 787–792 (2002)
- 18.161 M.J. Treacy, T.W. Ebbesen, J.M. Gibson: Exceptionally high Young's modulus observed for individual carbon nanotubes, Nature **381**, 678–680 (1996)
- 18.162 P. Poncharal, Z.L. Wang, D. Ugarte, W.A. de Heer: Electrostatic deflections and electromechanical resonances of carbon nanotubes, Science **283**, 1513–1516 (1999)
- 18.163 M.F. Yu, O. Lourie, M.J. Dyer, K. Moloni, T.F. Kelley, R.S. Ruoff: Strength and breaking mechanism of multiwalled carbon nanotubes under tensile load, Science **287**, 637–640 (2000)
- 18.164 T.W. Ebbesen, H.J. Lezec, H. Hiura, J.W. Bennett, H.F. Ghaemi, T. Thio: Electrical conductivity of individual carbon nanotubes, Nature **382**, 54–56 (1996)
- 18.165 P. Kim, L. Shi, A. Majumdar, P.L. McEuen: Thermal transport measurements of individual multiwalled nanotubes, Phys. Rev. Lett. **87**, 215502 (2001)
- 18.166 W.J. Liang, M. Bockrath, D. Bozovic, J.H. Hafner, M. Tinkham, H. Park: Fabry-Perot interference in a nanotube electron waveguide, Nature **411**, 665–669 (2001)
- 18.167 X.B. Zhang, D. Bernaerts, G.V. Tendeloo, S. Amelinckx, J.V. Landuyt, V. Ivanov, J.B. Nagy, P. Lambin, A.A. Lucas: The texture of catalytically grown coil-shaped carbon nanotubules, Europhys. Lett. **27**, 141–146 (1994)
- 18.168 X.Y. Kong, Z.L. Wang: Spontaneous polarization-induced nanohelices, nanosprings, and nanorings of piezoelectric nanobelts, Nano. Lett. **3**, 1625–1631 (2003)
- 18.169 S.V. Golod, V.Y. Prinz, V.I. Mashanov, A.K. Gutakovskiy: Fabrication of conducting GeSi/Si micro- and nanotubes and helical microcoils, Semicond. Sci. Technol. **16**, 181–185 (2001)
- 18.170 L. Zhang, E. Deckhardt, A. Weber, C. Schönenberger, D. Grützmacher: Controllable fabrication of SiGe/Si and SiGe/Si/Cr helical nanobelts, Nanotechnology **16**, 655–663 (2005)
- 18.171 L. Zhang, E. Ruh, D. Grützmacher, L.X. Dong, D.J. Bell, B.J. Nelson, C. Schönenberger: Anomalous coiling of SiGe/Si and SiGe/Si/Cr helical nanobelts, Nano Lett. **6**, 1311–1317 (2006)
- 18.172 D.J. Bell, L.X. Dong, B.J. Nelson, M. Golling, L. Zhang, D. Grützmacher: Fabrication and characterization of three-dimensional InGaAs/GaAs nanosprings, Nano Lett. **6**, 725–729 (2006)
- 18.173 D.J. Bell, Y. Sun, L. Zhang, L.X. Dong, B.J. Nelson, D. Grützmacher: Three-dimensional nanosprings for electromechanical sensors, Sens. Actuat. A-Physical **130**, 54–61 (2006)
- 18.174 R. Martel, T. Schmidt, H.R. Shea, T. Herte, P. Avouris: Single- and multi-wall carbon nanotube field-effect transistors, Appl. Phys. Lett. **73**, 2447–2449 (1998)
- 18.175 N.R. Franklin, Y.M. Li, R.J. Chen, A. Javey, H.J. Dai: Patterned growth of single-walled carbon nanotubes on full 4-inch wafers, Appl. Phys. Lett. **79**, 4571–4573 (2001)
- 18.176 T. Rueckes, K. Kim, E. Joselevich, G.Y. Tseng, C.-L. Cheung, C.M. Lieber: Carbon nanotube-based non-volatile random access memory for molecular computing science, Science **289**, 94–97 (2000)
- 18.177 A. Subramanian, B. Vikramaditya, L.X. Dong, D.J. Bell, B.J. Nelson: Micro and Nanorobotic Assembly Using Dielectrophoresis. In: *Robotics: Science and Systems I*, ed. by S. Thrun, G.S. Sukhatme, S. Schaal, O. Brock (MIT Press, Cambridge 2005) pp. 327–334
- 18.178 C.K.M. Fung, V.T.S. Wong, R.H.M. Chan, W.J. Li: Dielectrophoretic batch fabrication of bundled carbon nanotube thermal sensors, IEEE Trans. Nanotech. **3**, 395–403 (2004)
- 18.179 T. Fukuda, F. Arai, L.X. Dong: Assembly of nanodevices with carbon nanotubes through nanorobotic manipulations, Proc. IEEE **91**, 1803–1818 (2003)
- 18.180 E.W. Wong, P.E. Sheehan, C.M. Lieber: Nanobeam mechanics: elasticity, strength, and toughness of nanorods and nanotubes, Science **277**, 1971–1975 (1997)
- 18.181 M.R. Falvo, G.J. Clary, R.M. Taylor, V. Chi, F.P. Brooks, S. Washburn, R. Superfine: Bending and buckling of carbon nanotubes under large strain, Nature **389**, 582–584 (1997)
- 18.182 H.W.C. Postma, A. Sellmeijer, C. Dekker: Manipulation and imaging of individual single-walled carbon nanotubes with an atomic force microscope, Adv. Mater. **12**, 1299–1302 (2000)
- 18.183 T. Hertel, R. Martel, P. Avouris: Manipulation of individual carbon nanotubes and their interaction with surfaces, J. Phys. Chem. B **102**, 910–915 (1998)
- 18.184 P. Avouris, T. Hertel, R. Martel, T. Schmidt, H.R. Shea, R.E. Walkup: Carbon nanotubes: nanomechanics, manipulation, and electronic devices, Appl. Surf. Sci. **141**, 201–209 (1999)
- 18.185 M. Ahlskog, R. Tarkiainen, L. Roschier, P. Hakonen: Single-electron transistor made of two crossing multiwalled carbon nanotubes and its noise properties, Appl. Phys. Lett. **77**, 4037–4039 (2000)

- 18.186 M.R. Falvo, R.M.I. Taylor, A. Helser, V. Chi, F.P.J. Brooks, S. Washburn, R. Superfine: Nanometre-scale rolling and sliding of carbon nanotubes, *Nature* **397**, 236–238 (1999)
- 18.187 B. Bhushan, V.N. Koinkar: Nanoindentation Hardness Measurements Using Atomic-Force Microscopy, *Appl. Phys. Lett.* **64**, 1653–1655 (1994)
- 18.188 P. Vettiger, G. Cross, M. Despont, U. Drechsler, U. Durig, B. Gotsmann, W. Hablerle, M.A. Lantz, H.E. Rothuizen, R. Stutz, G.K. Binnig: The “millipede” – Nanotechnology entering data storage, *IEEE Trans. Nanotechnol.* **1**, 39–55 (2002)
- 18.189 L.X. Dong: Nanorobotic manipulations of carbon nanotubes. Ph.D. Thesis (Nagoya University, Nagoya 2003)
- 18.190 J.H. Hafner, C.-L. Cheung, T.H. Oosterkamp, C.M. Lieber: High-yield assembly of individual single-walled carbon nanotube tips for scanning probe microscopies, *J. Phys. Chem. B* **105**, 743–746 (2001)
- 18.191 L.X. Dong, F. Arai, T. Fukuda: Electron-beam-induced deposition with carbon nanotube emitters, *Appl. Phys. Lett.* **81**, 1919–1921 (2002)
- 18.192 L.X. Dong, F. Arai, T. Fukuda: 3D nanorobotic manipulations of multi-walled carbon nanotubes, *Proc. of 2001 IEEE Int. Conf. on Robotics and Automation (ICRA2001)* (IEEE, Piscataway 2001) pp. 632–637
- 18.193 L.X. Dong, F. Arai, T. Fukuda: Destructive constructions of nanostructures with carbon nanotubes through nanorobotic manipulation, *IEEE/ASME Trans. Mechatron.* **9**, 350–357 (2004)
- 18.194 J. Cumings, P.G. Collins, A. Zettl: Peeling and sharpening multiwall nanotubes, *Nature* **406**, 58 (2000)
- 18.195 J. Cumings, A. Zettl: Low-friction nanoscale linear bearing realized from multiwall carbon nanotubes, *Science* **289**, 602–604 (2000)
- 18.196 A. Kis, K. Jensen, S. Aloni, W. Mickelson, A. Zettl: Interlayer forces and ultralow sliding friction in multiwalled carbon nanotubes, *Phys. Rev. Lett.* **97**, 025501 (2006)
- 18.197 L.X. Dong, F. Arai, T. Fukuda: Nanoassembly of carbon nanotubes through mechanochemical nanorobotic manipulations, *Jpn. J. Appl. Phys.* **42**, 295–298 (2003), Part 1
- 18.198 L. Chico, V.H. Crespi, L.X. Benedict, S.G. Louie, M.L. Cohen: Pure carbon nanoscale devices: Nanotube heterojunctions, *Phys. Rev. Lett.* **76**, 971–974 (1996)
- 18.199 Z. Yao, H.W.C. Postma, L. Balents, C. Dekker: Carbon nanotube intramolecular junctions, *Nature* **402**, 273–276 (1999)
- 18.200 H.W.C. Postma, T. Teepen, Z. Yao, M. Grifoni, C. Dekker: Carbon nanotube single-electron transistors at room temperature, *Science* **293**, 76–79 (2001)
- 18.201 M.S. Fuhrer, J. Nygård, L. Shih, M. Forero, Y.-G. Yoon, M.S.C. Mazzoni, H.J. Choi, J. Ihm, S.G. Louie, A. Zettl, P.L. McEuen: Crossed nanotube junctions, *Science* **288**, 494–497 (2000)
- 18.202 T. Rueckes, K. Kim, E. Joselevich, G.Y. Tseng, C.-L. Cheung, C.M. Lieber: Carbon nanotube-based nonvolatile random access memory for molecular computing science, *Science* **289**, 94–97 (2000)
- 18.203 A.G. Rinzier, J.H. Hafner, P. Nikolaev, L. Lou, S.G. Kim, D. Tománek, P. Nordlander, D.T. Colbert, R.E. Smalley: Unraveling nanotubes: field emission from an atomic wire, *Science* **269**, 1550–1553 (1995)
- 18.204 S.C. Minne, G. Yaralioglu, S.R. Manalis, J.D. Adams, J. Zesch, A. Atalar, C.F. Quate: Automated parallel high-speed atomic force microscopy, *Appl. Phys. Lett.* **72**, 2340–2342 (1998)
- 18.205 G.D. Skidmore, E. Parker, M. Ellis, N. Sarkar, R. Merkle: Exponential assembly, *Nanotechnology* **11**, 316–321 (2001)
- 18.206 H.J. Dai, J.H. Hafner, A.G. Rinzier, D.T. Colbert, R.E. Smalley: Nanotubes as nanoprobe tips in scanning probe microscopy, *Nature* **384**, 147–150 (1996)
- 18.207 J.H. Hafner, C.L. Cheung, C.M. Lieber: Growth of nanotubes for probe microscopy tips, *Nature* **398**, 761–762 (1999)
- 18.208 L.X. Dong, B.J. Nelson, T. Fukuda, F. Arai: Towards Nanotube Linear Servomotors, *IEEE Trans. Autom. Sci. Eng.* **3**, 228–235 (2006)
- 18.209 Y.H. Gao, Y. Bando: Carbon nanothermometer containing gallium, *Nature* **415**, 599 (2002)
- 18.210 L.X. Dong, X.Y. Tao, L. Zhang, B.J. Nelson, X.B. Zhang: Nanorobotic spot welding: Controlled metal deposition with attogram precision from Copper-filled carbon nanotubes, *Nano Lett.* **7**, 58–63 (2007)
- 18.211 S.W. Lee, D.S. Lee, R.E. Morjan, S.H. Jhang, M. Sveningsson, O.A. Nerushev, Y.W. Park, E.E.B. Campbell: A three-terminal carbon nanorelay, *Nano Lett.* **4**, 2027–2030 (2004)
- 18.212 A.M. Fennimore, T.D. Yuzvinsky, W.-Q. Han, M.S. Fuhrer, J. Cumings, A. Zettl: Rotational actuators based on carbon nanotubes, *Nature* **424**, 408–410 (2003)
- 18.213 A. Subramanian, L.X. Dong, J. Tharian, U. Sennhauser, B.J. Nelson: Batch fabrication of carbon nanotube bearings, *Nanotechnology* **18**, 075703 (2007)
- 18.214 P. Kim, C.M. Lieber: Nanotube nanotweezers, *Science* **286**, 2148–2150 (1999)
- 18.215 L.X. Dong, A. Subramanian, D. Hugentobler, B.J. Nelson, Y. Sun: Nano Encoders based on Vertical Arrays of Individual Carbon Nanotubes, *Adv. Robot.* **20**, 1281–1301 (2006)
- 18.216 I. Asimov: *Fantastic Voyage* (Bantam Books, New York 1966)
- 18.217 R.A. Freitas: *Nanomedicine, Volume I: Basic Capabilities* (Landes Bioscience, Austin 1999)

Sensing and Perception

Part C

Part C Sensing and Perception

Ed. by Henrik I. Christensen

19 Force and Tactile Sensors

Mark R. Cutkosky, Stanford, USA
Robert D. Howe, Cambridge, USA
William R. Provancher, Salt Lake City, USA

20 Inertial Sensors, GPS, and Odometry

Gregory Dudek, Montreal, Canada
Michael Jenkin, Toronto, Canada

21 Sonar Sensing

Lindsay Kleeman, Monash, Australia
Roman Kuc, New Haven, USA

22 Range Sensors

Robert B. Fisher, Edinburgh, UK
Kurt Konolige, Menlo Park, USA

23 3-D Vision and Recognition

Kostas Daniilidis, Philadelphia, USA
Jan-Olof Eklundh, Stockholm, Sweden

24 Visual Servoing and Visual Tracking

François Chaumette, Rennes, France
Seth Hutchinson, Urbana, USA

25 Multisensor Data Fusion

Hugh Durrant-Whyte, Sydney, Australia
Thomas C. Henderson, Salt Lake City, USA

Part C of the Handbook is concerned with sensing and estimation. The world is not deterministic and, in addition, models of our robots are not perfect. Sensing is introduced to allow compensation for model approximations, and to estimate the layout of the environment to facilitate planning and execution of tasks.

Sensing is here considered the transformation of physical entities such as contact, force, distance, light intensity, etc. into an internal computer representation. Perception is the extraction of key properties from the sensory data and integration of sensory information over time. Perception is typically a data compression task in terms of extraction of the characteristics that allow recognition, tracking, and description of particular properties in the world, which are required to allow a robot to execute its tasks. This part of the Handbook covers the most typical sensor modalities used in robotics and the basic processes associated with these sensors.

Robotics is often paraphrased to be the intelligent coupling of perception and action. As such Part C complements Part B of the handbook and the following parts are all examples of such couplings in terms of sensing and mechanisms with associated intelligence.

This part is building on the foundations (Part A) and here in particular on Chap. 4, Sensing and Estimation. The material presented in this part is for obvious reasons tightly coupled to later chapters on systems and applications. Chapter 30 (Haptics) has a close coupling to the chapter on force and tactile sensing. Chapter 36 on world modeling is building on sensor models and the fusion methods presented in Chap. 25. The same applies for Chap. 37 on Simultaneous Localization and Mapping. Chapter 62 presents results from neuroscience and how it can be applied to robotics and has a strong coupling to many different sensory modalities and also integration of these into internal representations. Finally, Chap. 63 presents methods in human visual perception that are of relevance to robotics. These methods are in many ways the basis for techniques used in computer vision as discussed in Chaps. 23 and 24.

Part C contains chapters related to all the main sensory modalities from touch and proprioception to ranging and visual perception.

Chapter 19, Force and Tactile, discusses methods for detection of contact with objects in the external environment and also force as a reaction to acceleration and/or contact with external objects. Force sensing is essential to many high-precision assembly tasks, but increasingly also for interaction with nonrigid objects as, for example, experienced in medical applications. Contact and force sensing

is thus essential for many grasping and manipulation tasks.

Chapter 20, Inertial Sensors, GPS, and Odometry, covers methods for estimation of ego motion based on direct estimation from acceleration forces, shaft/wheel encoders or through exploitation of external reference systems such as GPS. Robot systems experience drift and slippage due to poor contact, but also due to material variations due to temperature etc. A fundamental capability is thus use of proprioception to estimate ego motion and internal state as a basis for decision making and action generation.

Chapter 21, Sonar Sensing, covers ranging and object detection based on sound information. Sound propagates significantly slower than light and as such allows use of electronics that it easier/cheaper to manufacture. This in part explains why much early work on mapping and position estimation for mobile platforms was based on ultrasonic sonars. Sonar is still a dominant modality for underwater applications and it often offers an economically attractive solution for basic mapping.

Chapter 22, Range Sensors, covers use of distance estimation based on light. Recently the use of laser radar has seen a significant increase in popularity. Laser-based sensors are active sensors in the meaning that the system emits energy of some form into the environment and measures the returned information, which allow use of matched techniques which simplifies the data-association problem and enable use in more noisy settings. Ranging can also be performed based on multiocular systems such as a stereo pair of cameras. The different methods for active and passive ranging are presented in this chapter.

Chapter 23, 3-D Vision and Recognition, covers 3-D modeling and recognition of objects based on image data. There is no doubt that vision is one of the most flexible and compelling sensory modalities. It has a significant potential for future applications. The flexibility poses, on the other hand, also a challenge as the detection of objects in cluttered environments can be very difficult. The chapter presents methods for visual modeling of objects and the main steps and methods for recognition of objects.

Chapter 24, Visual Servoing and Visual Tracking, presents the main methods for tracking and control based on image data. Tracking of objects is essential in many applications either for interception or avoidance. Visual servoing addresses the problem of control to arrive at a goal location often associated with grasping of an object. The main techniques for tracking are presented

and the three major methods for visual servoing are introduced.

Chapter 25, Sensor Fusion, addresses the problem of integration of information across space and time. Sensor data are contaminated by noise in various forms and often there is a need to integrate data from multiple sensors or across time to provide robust and accurate estimates. This chapter presents the main techniques for sensory fusion and provides mathematical frameworks that are utilized in several of the following chapters.

Due to changes in materials and electronics, the area of sensors is changing rapidly. Within the last decade major changes in terms of laser ranging and visual sen-

sors have been fielded. In addition computer resources have changed dramatically to allow advanced processing of data to generate advanced perception systems. The availability of cheap, robust, and accurate sensing is also impacting other fields of robotics. Earlier robots were built to have a maximum of stiffness to avoid sensing. Today flexible robots are relying on sensors in the internal control loop to achieve hyperaccuracy at a reduced cost. There is no doubt that this is a trend that will continue as flexible robots can be made safer, cheaper and more efficient. The integration of sensing and actuation, especially with new nanotechnology, is going to change robotics significantly over the next decade.

Force and Tactile Sensors

Mark R. Cutkosky, Robert D. Howe, William R. Provancher

This chapter provides an overview of force and tactile sensing, with the primary emphasis placed on tactile sensing. We begin by presenting some basic considerations in choosing a tactile sensor and then review a wide variety of sensor types, including proximity, kinematic, force, dynamic, contact, skin deflection, thermal, and pressure sensors. We also review various transduction methods, appropriate for each general sensor type. We consider the information that these various types of sensors provide in terms of whether they are most useful for manipulation, surface exploration or being responsive to contacts from external agents.

Concerning the interpretation of tactile information, we describe the general problems and present two short illustrative examples. The first involves *intrinsic* tactile sensing, i. e., estimating contact locations and forces from force sensors. The second involves contact pressure sensing, i. e., estimating surface normal and shear stress distributions from an array of sensors in an elastic skin.

We conclude with a brief discussion of the challenges that remain to be solved in packaging and manufacturing damage-tolerant tactile sensors.

19.1 Sensor Types	456
19.1.1 Proprioceptive and Proximity Sensing	456
19.1.2 Other Contact Sensors	458
19.1.3 Kinematic Sensors	459
19.1.4 Force and Load Sensing	459
19.1.5 Dynamic Tactile Sensors	460
19.1.6 Array Sensors	461
19.2 Tactile Information Processing	464
19.2.1 Tactile Information Flow: Means and Ends of Tactile Sensing ..	464
19.2.2 Solid Mechanics and Deconvolution	466
19.2.3 Curvature and Shape Information ..	469
19.2.4 Object and Surface Identification ...	469
19.2.5 Active Sensing Strategies	469
19.2.6 Dynamic Sensing and Event Detection	470
19.2.7 Integration of Thermal and Other Sensors	470
19.3 Integration Challenges	471
19.4 Conclusions and Future Developments	471
References	471

Tactile sensing has been a component of robotics for roughly as long as vision. However, in comparison to vision, for which great strides have been made in terms of hardware and software and which is now widely used in industrial and mobile robot applications, tactile sensing always seems to be *a few years away* from widespread utility. Therefore, before reviewing the technologies and approaches available it is worthwhile to consider some basic questions:

- How important is tactile sensing?
- What is it useful for?
- Why does it remain comparatively undeveloped?

In Nature, tactile sensing is an essential survival tool. Even the simplest creatures are endowed with large numbers of mechanoreceptors for exploring and responding to various stimuli. In humans, tactile sensing is indispensable for three distinct kinds of activities: manipulation, exploration, and response. The importance of tactile sensing for manipulation is most evident for fine motor tasks. When we are chilled, tasks like buttoning a shirt can become an exercise in frustration. The problem is primarily a lack of sensing; our muscles, snug in our coat sleeves, are only slightly affected but our cutaneous mechanoreceptors are anesthetized and we become clumsy. For exploration, we continually as-

simulate tactile information about materials and surface properties (e.g., hardness, thermal conductivity, friction, roughness) to help us identify objects. We may have difficulty distinguishing real leather from synthetic leather by sight, but not by touch. Finally, the importance of tactile response, whether to a gentle touch or an impact, is seen in the damage that patients with peripheral neuropathy (e.g., as a complication of diabetes) accidentally do to themselves.

As Fig. 19.1 indicates, the same functional categories apply to robots. However, in comparison to animals, with thousands of mechanoreceptors per square centimeter of skin, even the most sophisticated robots are impoverished. One reason for the slow development of tactile sensing technology as compared to vision is that there is no tactile analog to the charge-coupled device (CCD) or complementary metal–oxide–semiconductor (CMOS) optical array. Instead, tactile sensors elicit information through physical interaction. They must be incorporated into skin surfaces with compliance, to conform locally to surfaces, and with adequate friction to handle objects securely. The sensors and skin must also be robust enough to survive repeated impacts and abrasions. And unlike the image plane in a camera, tactile sensors must be distributed over the robot appendages, with particularly high concentrations in areas such as the fingertips. The wiring of tactile sensors is consequently another formidable challenge.

19.1 Sensor Types

This section outlines five main types of sensors: proprioceptive, kinematic, force, dynamic tactile, and array tactile sensors. A basic review of the first three of these is provided along with contact sensors that provide thermal or material composition data. However, greater emphasis is placed on tactile sensors that provide mechanoreception. Table 19.1 provides an overview of these tactile sensors. When considering tactile sensors, it is useful to begin by considering the fundamental physical quantities that can only be sensed through contact with the environment. The most important quantities measured with touch sensors are *shape* and *force*. Each of these may be measured either as an average quantity for some part of the robot or as a spatially resolved, distributed quantity across a contact area. In this chapter we follow the convention of studies of the human sense of touch and use the term *touch sensing* to refer to the combination of these two modes. Devices that measure

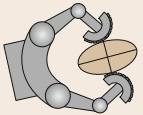
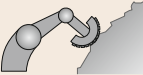
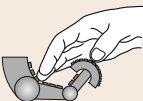
	<i>Manipulation:</i> Grasp force control; contact locations and kinematics; stability assessment.
	<i>Exploration:</i> Surface texture, friction and hardness; thermal properties; local features.
	<i>Response:</i> Detection and reaction to contacts from external agents.

Fig. 19.1 Uses of tactile sensing in robotics

Nonetheless, considerable progress in tactile sensor design and deployment has been made over the last 20 years. In the following sections we review the main functional classes of tactile sensors and discuss their relative strengths and limitations. Looking ahead, new fabrication techniques offer the possibility of artificial skin materials with integrated sensors and local processing for interpreting sensor signals and communicating over a common bus to reduce wiring.

There is an extensive literature describing touch sensing research. Recent general reviews include [19.1] and [19.2], and these cite several older reviews from the 1980s and 1990s.

an average or resultant quantity are sometimes referred to as *internal* or *intrinsic* sensors. The basis for these sensors is force sensing, which precedes the discussion of tactile array sensors.

19.1.1 Proprioceptive and Proximity Sensing

Proprioceptive sensing refers to sensors that provide information about the net force or motion of an appendage, analogous to receptors that provide information in humans about tendon tensions or joint movements. Generally speaking the primary source for spatial proprioceptive information on a robot is provided by joint angle and force-torque sensors. Since joint angle sensors such as potentiometers, encoders, and resolvers are well-established technologies, they do not warrant discussion here. Instead, a brief review of proximity sensing via whiskers and antennae as well as noncontact proximity

Table 19.1 Tactile sensor modalities and common transduction types

Sensor modality	Sensor type and attributes	Advantages	Disadvantages
Normal pressure			
	Piezoresistive array [19.3–8]		
	<ul style="list-style-type: none"> • Array of piezoresistive junctions • Embedded in an elastomeric skin • Cast or screen printed 	<ul style="list-style-type: none"> • Simple signal conditioning • Simple design • Suitable for mass production 	<ul style="list-style-type: none"> • Temperature sensitive • Frail • Signal drift and hysteresis
	Capacitive array [19.9–13]		
	<ul style="list-style-type: none"> • Array of capacitive junctions • Row and column electrodes separated by elastomeric dielectric 	<ul style="list-style-type: none"> • Good sensitivity • Moderate hysteresis, depending on construction 	<ul style="list-style-type: none"> • Complex circuitry
	Piezoresistive MEMS array [19.14, 15]		
	<ul style="list-style-type: none"> • Silicon micromachined array with doped silicon strain-gauged flexures 	<ul style="list-style-type: none"> • Suitable for mass production 	<ul style="list-style-type: none"> • Frail
	Optical [19.16]		
	<ul style="list-style-type: none"> • Combined tracking of optical markers with a constitutive model 	<ul style="list-style-type: none"> • No interconnects to break 	<ul style="list-style-type: none"> • Requires PC for computing applied forces
Skin deformation			
	Optical [19.17]		
	<ul style="list-style-type: none"> • Fluid-filled elastomeric membrane • Tracking of optical markers inscribed on membrane coupled with energy minimization algorithm 	<ul style="list-style-type: none"> • Compliant membrane • No electrical interconnects to be damaged 	<ul style="list-style-type: none"> • Complex computations • Hard to customize sensor
	Magnetic [19.18]		
	<ul style="list-style-type: none"> • Array of Hall-effect sensors 		<ul style="list-style-type: none"> • Complex computations • Hard to customize sensor
	Resistive tomography [19.19]		
	<ul style="list-style-type: none"> • Array of conductive rubber traces as electrodes 	<ul style="list-style-type: none"> • Robust construction 	<ul style="list-style-type: none"> • Ill-posed inverse problems
	Piezoresistive (curvature) [19.20]		
	<ul style="list-style-type: none"> • Employs an array of strain gauges 	<ul style="list-style-type: none"> • Directly measure curvature 	<ul style="list-style-type: none"> • Frailty of electrical interconnects • Hysteresis

Table 19.2 (cont.)

Dynamic tactile sensing		
Piezoelectric (stress rate) [19.21–23]		
● PVDF (polyvinylidene difluoride) embedded in elastomeric skin	● High bandwidth	● Frailty of electrical junctions
Skin acceleration [19.23, 24]		
● Commercial accelerometer affixed to robot skin	● Simple	● No spatially distributed content
		● Sensed vibrations tend to be dominated by structural resonant frequency

sensing is provided. Force-torque sensors are discussed in greater detail in Sect. 19.1.4.

Whisker and Antenna Sensors

Whisker or antenna sensors are in essence a hybrid of proprioceptive and tactile information. This form of sensing was first explored in the early 1990s, for example, *Russell* [19.25] developed a whisker sensor with a base angle sensor and tip contact sensor that was attached to a robot arm to explore its environment. Another example by *Kaneko* et al. [19.26] is one of the earliest examples of active antenna sensing. *Kaneko* et al. affixed a rigid spring steel antenna to a one-degree-of-freedom (1-DOF) rotating axis used to sweep the antenna from side to side similar to the method an insect would employ. The sweeping motion, in combination with a joint angle sensor and torque sensor, was used to assess encountered contacts. *Clements* and *Rahn* [19.27] took a similar approach to *Kaneko*, but added an extra degree of freedom to the sweeping pattern of their whisker. *Clements* and *Rahn* used a motor-driven gimbal to drive their spring steel whisker in two DOFs to explore objects. *Cowan* et al. [19.28] used a multisegmented piezoresistive antenna to aid a bio-inspired insect hexapod robot in a wall-following control task.

For many animals, whiskers or antennae provide an extremely accurate combination of contact sensing and proprioceptive information, for example, cockroaches can steer themselves along curved walls at 20 body-lengths per second using only the position and rate information that they obtain from their antennae. Other insects, such as arthropods use numerous small hair sensors on their exoskeleton to localize contacts.

Proximity

While proximity sensing does not strictly fall under the category of tactile sensing, a number of researchers have employed various proximity sensors for the application of collision detection between a robot arm and the environment and thus we briefly review these technologies here. Three primary sensor technologies which include capacitive, infrared (IR) optical, and ultrasonic sensors have been used in this application. *Vranish* et al. developed an early capacitive sensor for collision avoidance between the environment and a grounded robot arm [19.29]. Examples of distributed IR emitter–detector pairs utilized within artificial skin for the purposes of proximity sensing have been presented by *Lumelsky*’s research group [19.30, 31]. A more recent design using optical fibers is reported in [19.32]. Other researchers have developed robot skins that include both distributed ultrasonic and IR optical sensors for the purposes of collision avoidance [19.33]. *Wegerif* and *Rosinski* provide a comparison of the performance of all three of these proximity sensing technologies [19.34]. For a more detailed review of this variety of sensors, see Chap. 21 on sonar sensing and Chap. 22 on range sensors.

19.1.2 Other Contact Sensors

There are a variety of other contact-based sensors that are capable of discerning object properties such as electromagnetic characteristics, density (via ultrasound), or chemical composition (cf. animals, senses of taste and smell). While this is beyond the scope of the current chapter, Chap. 60 on biologically inspired robots briefly discusses biologically inspired chemical sensors related

to smell and taste. For completeness, thermal sensors and material composition sensors are also briefly discussed below.

Thermal Sensors

Thermal sensing can be used to determine the material composition of an object as well as to measure surface temperatures. Since most objects in the environment are at about the same (*room*) temperature, a temperature sensor that contains a heat source can detect the rate at which heat is absorbed by an object. This provides information about the heat capacity of the object and the thermal conductivity of the material from which it is made, making it easy, for example, to distinguish metals from plastics.

Buttazzo et al. [19.35] note that the piezoelectric polymer used in their tactile sensing system is also strongly pyroelectric, and use a superficial layer as a thermal sensor. Other sensors use thermistors as transducers, with Siegel et al. [19.36] reporting a 4×4 array and Russell [19.37] a 2×10 array. Some systems purposely provide an internal temperature reference and use the temperature differential from the environment as a means of finding contacts [19.38, 39]. However, objects with a temperature the same as the reference will not be detected. Most of these sensors have a relatively thick outer skin covering the heat-sensitive elements, thus protecting delicate components and providing a conformal surface at the expense of slower response time.

A more recent example of thermal sensing can be found in the work of Engel et al., who present a flexible tactile sensor design that includes integrated gold film heaters and a resistance temperature device (RTDs) on a polymer micromachined substrate [19.40]. While there is a high level of integration presented by Engel et al., these sensing elements still remain fragile, hence tradeoffs concerning construction, performance, and the protection of sensing elements in these systems remains an ongoing challenge.

Material Composition Sensors

There has been a little work on sensors for material composition. In analogy with the human senses of taste and smell, liquid- and vapor-phase chemical sensors could potentially determine the chemical composition of a surface ([19.41, 42]). Another sensing modality which provides information about material properties is electromagnetic field sensing, using devices such as eddy-current or Hall-effect probes to measure ferromagnetism or conductivity [19.43, 44].

19.1.3 Kinematic Sensors

Although they are not generally thought of as tactile sensors, sensors that detect the position of a limb can provide the robot with geometric information for manipulation and exploration, particularly when the limb also includes sensors that register contact events. Examples of such sensors include the ubiquitous joint angle encoders found in virtually all robots as well as potentiometers, resolvers, and other joint angle measuring devices. For limbs that do not undergo large rotations one can also embed flexible structures such as elements composed of piezoresistive ink, e.g., as used by Cowan et al. [19.28], as discussed in Sect. 19.1.1. Examples of combining information about joint angles with contact status sensors for manipulation include Kaneko's work [19.45] on the *posture changeability* of fingers.

19.1.4 Force and Load Sensing

Actuator Effort Sensors

For some actuators such as electric servo motors, a measure of the actuator effort can be obtained directly by measuring the motor current (typically using a sensing resistor in series with the motor and measuring the voltage drop across the sense resistor). However, because motors are typically connected to robot limbs via gearboxes with output/input efficiencies of 60% or less, it is usually much more accurate to measure the torque at the output of the gearbox. Solutions to this problem include shaft torque load cells (typically using strain gages) and mechanical structures at the robot joints whose deflections can be measured using electromagnetic or optical sensors. For cable- or tendon-driven arms and hands it is useful to measure the cable tension – both for purposes of compensating for friction in the drive-train and as a way of measuring the loads upon the appendage [19.46]. When fingers or arms make contact with objects in the environment, cable tension sensing becomes an alternative to endpoint load sensing for measuring components of the contact forces. Of course, only those components that produce significant torques can be measured with any accuracy. Section 15.3.2 (Robot Hands: Sensors) contains more details concerning cable tension measurement.

Force Sensors

When actuator effort sensors are not sufficient to measure the forces exerted by or on a robot appendage, discrete force sensors are typically utilized. These sensors are found most often at the base joint or wrist of

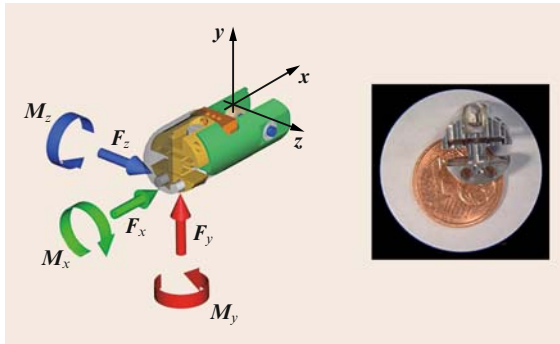


Fig. 19.2 Miniature fingertip force-torque sensor for a prosthetic hand (after [19.47] with permission)

a robot, but could be distributed throughout the links of a robot.

In principle, any type of multiaxis load cell could be used for manipulator force-torque sensing. However, the need for small, lightweight units with good static response eliminates many commercial sensors. The design of force sensors for mounting above the gripper at the wrist has received the most attention [19.48, 49], but fingertip sensors for dextrous hands have also been devised. Often these sensors are based on strain gauges mounted on a metal flexure [19.50, 51], which can be fairly stiff and robust. *Sinden* and *Boie* [19.52] propose a planar six-axis force-torque sensor based on capacitive measurements with an elastomer dielectric. Design considerations for force sensors include stiffness, hysteresis, calibration, amplification, robustness, and mounting. *Dario* et al. present an integrated fingertip for robotic hands: an integrated force sensing resistor (FSR) pressure array, piezoceramic bimorph dynamic sensor, and force-torque sensor [19.21]. More recently *Edin* et al. [19.47] have developed a miniature multi-axis fingertip force sensor (Fig. 19.2). For applications where immunity to electromagnetic noise is desirable, *Park* [19.53] presents a design for a robot fingertip with embedded fiber optic Bragg gratings, used as optical strain gages. *Bicchi* [19.54] and *Uchiyama* et al. [19.55] consider the optimal design of multiaxis force sensors in general.

It is interesting to note that more than just force information can be gained by the use of fingertip load sensors. Information from the force sensors can be combined with knowledge of fingertip geometry to estimate contact location, as implied in Fig. 19.3. This method of contact sensing is referred to as *intrinsic tactile sensing*, and was first presented by *Bicchi* et al. [19.56]. A comparison between intrinsic and extrinsic contact sensing

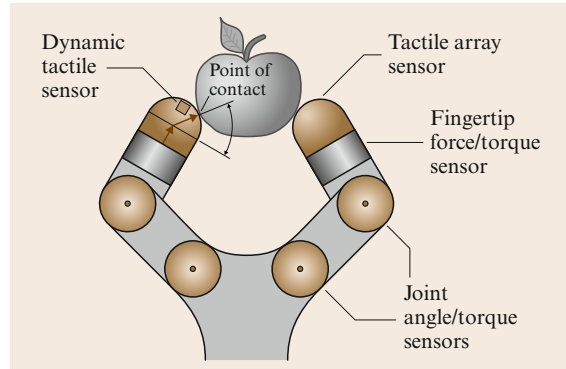


Fig. 19.3 Robot hand with fingertip force and tactile sensing. Information from the force sensors can be combined with knowledge of fingertip geometry to estimate contact location, referred to as *intrinsic tactile sensing*

(i.e., using distributed contact sensors) is presented by *Son* et al. [19.11]. This is also discussed in further detail in Sect. 19.2.1

19.1.5 Dynamic Tactile Sensors

Early special-purpose slip sensors based on displacement detected the motion of a moving element such as a roller or needle in the gripper surface (e.g., *Ueda* et al. [19.57]). A more recent approach uses a thermal sensor and a heat source: when the grasped object begins to slip, the previously warmed surface under the sensor moves away, causing a drop in surface temperature beneath the sensor. A noncontact optical approach uses correlation to reveal motion of the object surface [19.58]. A number of researchers have suggested using conventional arrays for slip detection, but the array resolution must be good and the scanning rate high to detect the motion of object features soon enough to prevent dropping the grasped object.

In a systematic investigation of the feasibility of using vibration to detect slip, *Rebman* and *Kallhammer* [19.59] used single elements from an array sensor to detect normal vibrations at the contact surface. *Dario* and *DeRossi* [19.60] and *Cutkosky* and *Howe* [19.61] note that piezoelectric polymer transducers located near the contact surface are very sensitive to vibrations and may be used for slip detection. *Howe* and *Cutkosky* [19.24] show that using a small accelerometer to sense minute vibrations of a compliant sensor skin is an effective means of detecting slip at its earliest stages. For hard objects held in metal grippers, acoustic emissions may reveal incipient slip [19.62, 63]. *Mor-*

rell [19.64] and Tremblay [19.65] investigated the use of slip sensors in grasp force control.

Buttazzo et al. [19.35] have built a texture-sensing *finger nail* as part of their anthropomorphic tactile sensing system. A piezoelectric element at the base of the rigid plastic nail produces a large signal as it is dragged over a textured surface. The stress rate sensor [19.22, 61, 66], the skin acceleration sensor [19.22, 24], and the induced vibration sensor [19.67] described above in the context of shape or slip sensing also respond to the small vibrations produced by sliding over fine surface textures. More recent adaptations of these sensors include piezoceramic bimorph dynamic sensors, with integrated FSR pressure array, and force-torque sensor [19.21]. Yamada et al. [19.68], who have developed a piezoelectric artificial skin, are able to distinguish between rolling and slip. Waldron et al. [19.23] describe a tactile sensor that integrates skin acceleration with a piezoelectric array for the application of teledermatology. Ellis has also investigated the use of a very high-resolution tactile array data for discriminating surface textures [19.69]. Omata and Terunuma present a sensor that measures changes in compliance by detecting changes in the resonant frequency of an active piezoelectric element [19.70].

19.1.6 Array Sensors

Array sensors can be subdivided into two primary categories: those that measure pressure and those that measure the deflection of the sensor skin. Tactile pressure arrays are by far more common. Pressure arrays tend to be relatively stiff and utilize a variety of transduction methods along with solid mechanics to calculate contact pressure distribution. Skin deformations/deflections in pressure arrays are on the order of 1–2 mm. On the other hand, skin deflection sensors are constructed in a manner that permits gross sensor skin deformation during contacts, which can be advantageous for grasp stability (see Chaps. 27 and 28).

Hundreds of designs for tactile array sensors have appeared in the literature in the last 15 years, and a few of them are suitable for use with dextrous hands. In terms of transducers, the fundamental requirement is to recover either the shape or pressure distribution across the contact unambiguously from beneath a compliant elastomer layer. This can be done by directly sensing shape [19.71, 72] or by sensing multiple components of subsurface strain [19.73, 74]. Solid mechanics models (Sect. 19.2.2) can then be used to determine the desired quantities. While the approach used in the models

developed to date is quite successful, they must be extended to three dimensions and to incorporate multiple components of strain. Since these models are complex, real-time execution must be addressed.

Contact Location Sensors

There have been a number of tactile sensors that solely provide contact location. Discrete switches placed on the outer chassis of mobile robots were quite common before the proliferation of ultrasonic sensors. Other researchers have presented work using discrete switch arrays. Some such sensors utilize a membrane switch design, found in keyboards. A visually transparent example of such a design is presented by Arai et al. for use in combination with a touch screen display [19.75]. Alternately, Griffin [19.76] shows a membran-switch-inspired contact switch array designed by W. Provancher that consists of flexible circuits separated by thin dielectric strips. Contacts are registered when pressure deflects one flexible circuit into the other, as shown in Fig. 19.4a. More recent work presented by Edin et al. [19.47]

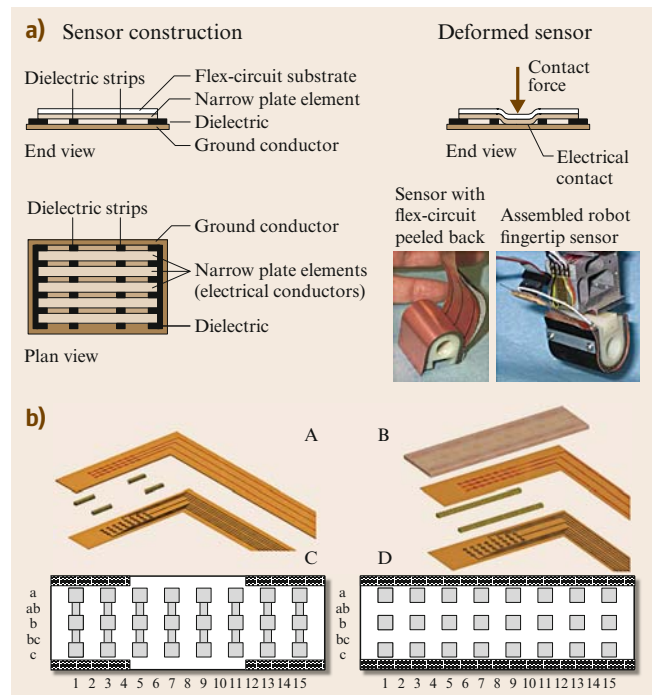


Fig. 19.4a,b Contact switch arrays fabricated from flexible printed circuits. (a) A simple 16×1 switch array used on the fingertip of a dextrous robot hand (drawn by W. Provancher). (b) Contact switch array embedded in the skin of a prosthetic hand (after [19.47] with permission)

shows a nice extension on this idea, with a two-dimensional switch array embedded in a prosthetic hand; see Fig. 19.4b. Some optical tactile sensors [19.77] have also been used primarily as contact location sensors. However, because they can also measure the magnitude of the contact force they are discussed under tactile arrays.

Pressure Sensing Arrays

Capacitive Pressure Sensing Arrays. Tactile pressure arrays are one of the oldest and most common tactile sensor types. Some of the earliest work in this area was done by *Fearing* [19.78] and *Fearing and Binford* [19.9]. These researchers produced capacitive tactile pressure arrays within a robot fingertip, suitable for dexterous manipulation. The sensor array consisted of overlapping row and column electrodes separated by an elastomeric dielectric, forming an array of capacitors. A change in capacitance results from compressing the dielectric between row–column plates at a particular intersection. The equation for capacitance, based on the physical parameters, is expressed as $C \approx (\epsilon A)/d$, where ϵ is the permittivity of the dielectric between the plates of the capacitor, A is the area of the plates, and d is the spacing between them. Compressing the dielectric between the capacitive plates thus reduces the plate spacing d , providing a linear response with displacement.

Through appropriate switching circuitry, a particular region of a sensor array can be isolated at a particular row–column intersection. Pressure is calculated by means of solid mechanics, as outlined by *Fearing* [19.79]. Examples of similar capacitive tactile arrays can be found in [19.10, 11] and commercially at [19.12]. More recently, *Shinoda's* group has investigated conductive fabric to form pairs of conforming, stacked capacitors more appropriate for covering larger areas than a robot hand [19.13]. The fabric electrodes in the stacked capacitive sensor elements are alternately separated by stiff or soft urethane foam, permitting simultaneous estimation of contact pressure and area within a single sensor element.

Piezoresistive Pressure Sensing Arrays. A variety of researchers have produced tactile sensor arrays that are piezoresistive in nature. These sensors generally either utilize a conductive rubber that is bulk molded or a piezoresistive ink that is generally patterned via screen printing or stamping. Each of these employ a conductive additive (typically carbon black) to create its conductive/piezoresistive behavior. However, because of the fragility and hysteresis that these sensor mor-

phologies exhibit, some researchers have also developed fabric-based piezoresistive sensors.

Russell [19.80] presented one of the first molded conductive rubber tactile sensor arrays composed of conductive rubber column and row electrodes with piezoresistive junctions. However, this sensor exhibited significant drift and hysteresis, which became the topic of later research to try to minimize these effects through proper selection of molding material [19.3]. These issues were never completely solved due to the hysteretic nature of elastomers, but this sensing approach remains attractive due to its ease of manufacturing. Hence it has continued to find applications in the appendages of humanoid robots where extreme accuracy is not required [19.81].

A number of researchers and companies have developed tactile sensors that utilize conductive (piezoresistive) ink, generally referred to as force sensitive resistors (FSRs). This is by far the most common, simplest, and most readily available means to incorporate tactile sensing via off-the-shelf discrete sensors (see Tekscan Flexiforce FSRs). However, to make highly integrated, dense sensor arrays, custom fabrication is necessary. Examples of such sensors are presented by *Papakostas et al.* [19.4] and *Dario et al.* [19.21]. To take this approach one step further, *Someya* [19.5] has produced robotic skin that employs patterned organic semiconductors for local amplification of the piezoresistive sensor array, printed on flexible polyimide film. However, despite being fabricated on a flexible substrate, these sensor arrays are still susceptible to bending fatigue.

Piezoresistive fabrics have been developed to address fatigue and fragility issues found in tactile arrays. Examples of these sensors are presented by *De Rossi et al.* [19.7], *Tognetti et al.* [19.8], and *Shimojo et al.* [19.6]. These sensors tend to be larger (i. e., have lower spacial resolution) and are utilized in applications such as the arms or legs of humanoid robots. Because this technology has the potential to replace ordinary cloth, it is a promising technology for applications in wearable computing or even smart clothing.

One final design that does not fall under the above fabrication categories is a sensor designed by *Kageyama et al.* that utilizes a piezoresistive conductive gel pressure array along with a multilevel contact switch array via variable contact resistance within sensor layers that they developed for use in humanoid robots [19.82].

MEMS Pressure Sensing Arrays. Micro-electromechanical (MEMS) technology is quite attractive for producing

highly integrated packaging for tactile sensing and associated interconnects and electronics. Early devices were produced in silicon through standard silicon micromachining techniques, including the silicon micromachined CMOS-compatible tactile array capable of measuring shear and normal force developed by Kane et al. [19.14]. These sensors functioned well on the laboratory desktop, but were too fragile to survive impact or harsh environments. More recently, researchers have applied MEMS techniques to produce sensor arrays appropriate for embedding within an elastomeric skin with flexible substrates. This has the advantage of improved durability because of the protective skin. Engel et al. have produced a tactile sensor with combined temperature and cantilever elements grown on a polyimide backing [19.40]. Polymer micromachining of the polyimide was performed to give the polyimide substrate preferential compliance. An interesting variation on this design is presented by Noda and Shimoyama [19.83]. These researchers also produced cantilever shear stress elements on their array sensor, however, they do this in a planar fabrication process to fabricate cantilevers with NiCr piezoresistors that are subsequently released from the fabrication substrate and made to stand up off the substrate like hairs by applying a magnetic field and subsequently fixing the cantilevered beams by covering them with Parylene-C (used for sensing

shear) [19.83]. Returning to the idea of using silicon load cells for tactile sensing, Valdastrì et al. [19.15] have developed a miniature MEMS silicon-based load cell that resembles a joystick and that is appropriate for embedding within an elastomeric skin (see Fig. 19.5). These sensors could be distributed beneath the skin surface to detect complex stress states in the elastomeric skin.

Skin Deflection Sensing

Brockett [19.84] was one of the first to propose the idea of using a deformable membrane robot fingertips. As noted by Shimoga and Goldenberg [19.85], there are several advantages to using deformable fingertips over more rigid robot fingertips, which include: (1) improved grasp stability, (2) reduced shock, and (3) reduced fatigue for embedded sensor elements. Early work on deformable fingertips includes Russell's work to create a compliant silicone rubber robot finger [19.80]. His sensor finger used a rigid backing and polyurethane foam to provide a restoring force for the sensor skin. An array of conductive rubber strain gauge elements and their associated interconnects were cut to the correct shape and bonded to the backside of the silicone rubber skin. The conductive rubber was simply silicone rubber, chosen with minimal mechanical hysteresis, mixed with graphite. The electrical interconnects of a given row tapped into the conductive rubber strain gauge at regular intervals along its length, thus subdividing it into several individual strain measurements, forming a potential divider. Russell shows results for both an 8×1 line array and an 8×5 array.

Nowlin used Bayesian algorithms to improve data interpretation of a deformable tactile sensor that used magnetic field sensing [19.72]. A 4×4 array of magnets were supported above paired Hall-effect sensors off a rigid base by individual fluid-filled balloons. Hall-effect sensors measure the strength of the local magnetic field and should therefore increase with closer proximity to the magnets; however, this is complicated by the proximity of neighboring magnets in the sensor array. Hence, noisy data from the Hall-effect sensors were combined using Bayesian algorithms to predict membrane deformation for a deformable fingertip [19.72].

Later Russell and Parkinson [19.19] developed an impedance tomographic tactile sensor capable of measuring skin deformation over an 8×5 array. This sensor was constructed with neoprene rubber and filled with distilled water. Row and column electrodes were made from copper and conductive rubber for the rigid substrate and neoprene skin, respectively. Like the capacitive

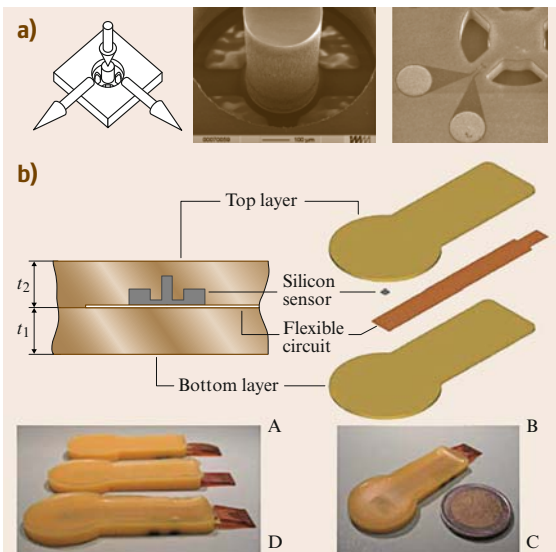


Fig. 19.5 (a) SEM micrographs of a MEMS three-axis tactile force sensor. (b) MEMS force sensor wire-bonded to a flex-circuit and embedded within a silicone rubber skin [19.15]

tactile sensors described above, this sensor utilized multiplexing electronics to reduce the number of electrical interconnects. Square-waveform driver electronics are used to estimate the resistance of a column of water formed between row and column elements, providing a signal that is proportional to the current skin height.

After suggesting the idea of using deformable tactile sensors, *Ferrier and Brocket* [19.17] implemented a tactile sensor which used optical tracking in combination with models of the sensors skin deformation to predict sensor fingertip skin deformations. The fingertip sensor consists of a tiny **CCD** camera focused on a 7×7 array of dots that are marked on the inside of the gel-filled silicone fingertip membrane. An algorithm is used to construct a 13×13 grid over the array of dots. This algorithm uses a combination of the position that is sensed by the **CCD** camera, which provides the location along a line radially outward from the focal point, and a mechanical model used to solve for the radial distance from the camera focal point based on energy minimization.

Provancher and Cutkosky present another design that uses piezoresistive strain gauges to measure fingertip membrane curvature directly [19.20]. The sensor is constructed on a polyimide substrate with commercial strain gauge strips (used to investigate strain gradients) bonded back to back to isolate bending strains, which are proportional to curvature. This curvature information is important for motion planning in dexterous manipulation. The authors present a mathematical model which uses a set of basis functions and least-squares minimization to calculate a curvature-space curve fit that is insensitive to sensor noise and capable of reconstruction of the deformed membrane shape. Results from a 11×1 line array sensor prototype are presented in Sect. 19.2.2.

Other Tactile Array Sensors

An early example of an optical tactile sensor was presented by *Maekawa et al.* [19.77], who used a single optical position sensing device (PSD) or a **CCD** camera array to detect the position of scattered light off of

a hemispherical optical waveguide fingertip with a silicone rubber cover. Light is scattered at the location of contact and, based on a simple model, multiple points of contact can be estimated. With a textured skin, the magnitude of the force can also be estimated, as the contact area grows in proportion to the pressure. However, an issue with fingertips that use a compliant skin covering a hard substrate is that adhesion between the two materials results in hysteresis. In addition, when the fingertip is dragged over a surface, the friction can produce a shift in the estimated contact position.

Another interesting tactile sensor uses vision to track an array of spherical markers embedded in a transparent elastomer to infer the stress state of the skin material due to applied forces [19.16]. This sensor is currently being commercialized under the tradename GelForce.

A number of sensors have been developed that monitor changes in acoustic energy reflected off of a skin surface or due to the distortion of an acoustic subsurface cavity. *Shinoda et al.* [19.86] present a sensor that looks at the change in reflected acoustic energy from an acoustic resonator chamber near the surface of the skin and its application for instantaneous friction measurement [19.87]. *Ando et al.* [19.88] present a more sophisticated ultrasound sensor that achieves 6-DOF displacement sensing via paired plate elements that utilize four ultrasound transducers per plate.

Several researchers have developed sensors with multiple sensing modalities. Examples of sensors that combine tactile and thermal sensing are presented by *Siegel et al.* [19.89] and *Castelli* [19.90]. *Shimizu et al.* [19.91] present a sensor for measuring force and the hardness of an object. This sensor actively uses pneumatic pressure to drive sensing elements into an object's surface.

Another interesting sensor includes a combination of tactile sensing and ultrasound imaging [19.92], in which *Methil et al.* have developed a haptic system for performing breast examinations that utilizes an optical waveguide tactile sensor.

19.2 Tactile Information Processing

19.2.1 Tactile Information Flow: Means and Ends of Tactile Sensing

In discussing the processing of tactile information we return first to the three main uses depicted in Fig. 19.1. For

manipulation, we require, foremost, information about contact locations and forces so that we can grasp objects securely and impart desired forces and motions to them. For exploration, we are concerned with obtaining and integrating information about the object, including the

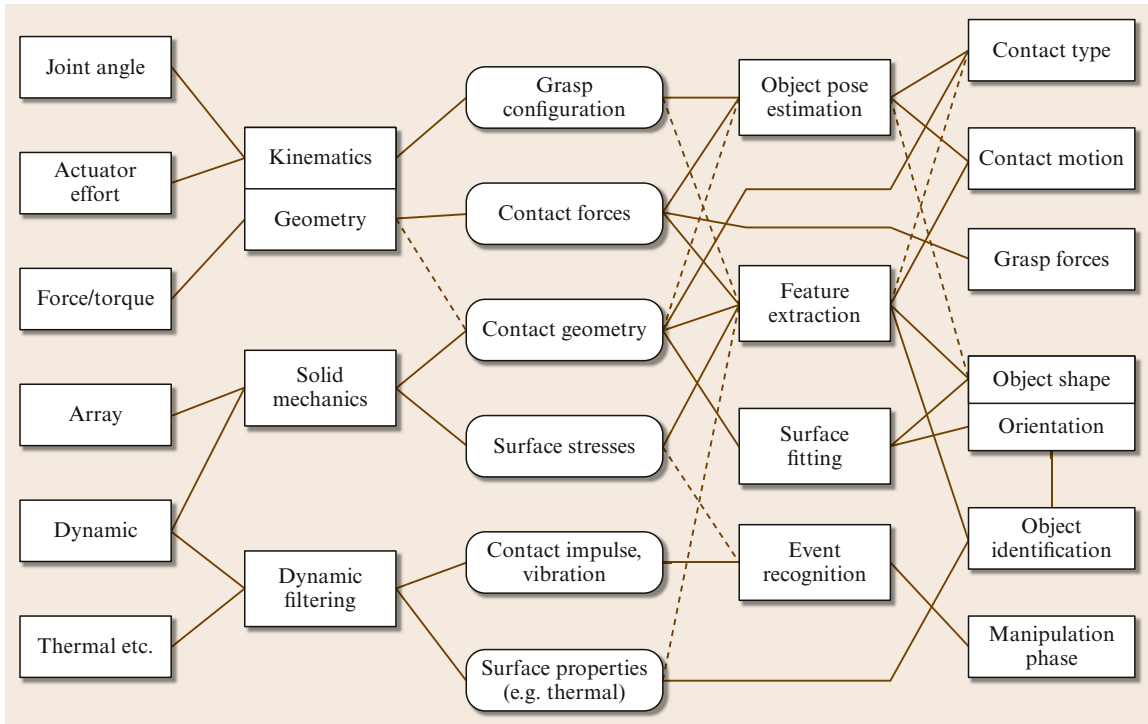


Fig. 19.6 Force and tactile sensor information flow and signal processing

local geometry, hardness, friction, texture, thermal conductivity, etc. For response, we are concerned especially with the detection of events, such as contacts produced by an external agent, and in assessing their types and magnitudes. The uses of information are often coupled; for example, we manipulate objects in order to explore them and we use the information obtained through object exploration to improve our ability to control forces and motions in manipulation. Recognizing contact events is also important for manipulation and exploration, as it is for response.

Figure 19.6 summarizes the general flow of information from each type of sensor reviewed in the previous section through primary sensed quantities to information provided for manipulation, exploration and response. A useful thought exercise is to consider exactly what information we use to perform a task such as turning a pen end over end between the fingers. We can easily perform this task with our eyes closed. What information are we using? We need to track the position and orientation of the pen and to monitor the forces that we impose on it to maintain stable manipulation. In other words, we need to know the *configuration* of our grasp, the *locations and movements of contacts* over the surfaces of

our fingers, the *magnitudes of grasp forces*, the *contact conditions* with respect to friction limits, etc. The same requirements apply for robots and are provided by the information flow in Fig. 19.6.

At the upper left corner of the figure, joint angles, combined with the forward kinematic model of the manipulator and knowledge of external link geometries, establish the positions and orientations of coordinate frames embedded in the fingertips. This information is needed to integrate local information about object shape, surface normal orientation, etc. so that the overall geometry and pose of the object can be determined.

Actuator effort sensors provide information about the resultant forces, using the Jacobian transpose: $J^T \mathbf{f} = \boldsymbol{\tau}$, where \mathbf{f} is an $n \times 1$ vector of external forces and moments taken with respect to a coordinate frame embedded in the appendage. J^T is the Jacobian transpose, mapping external forces and moments to joint torques and $\boldsymbol{\tau}$ is an $m \times 1$ vector of joint torques for a serial kinematic chain with m degrees of freedom. We require that the k -th column of J^T have elements that are relatively large compared to the overall condition number of J in order to provide an accurate measurement of

the k -th element of \mathbf{f} . Eberman and Salisbury [19.93] show that it is possible to measure contact force and location using only joint torque measurements if the manipulator has clean dynamics.

Alternatively, we can use a multiaxis force/torque sensor in the fingers, as indicated in Fig. 19.3, or robot wrist to obtain contact forces. This approach has the advantage of providing dynamic force signals with a higher signal-to-noise ratio because they are not masked by the inertias of the robot arm or fingers and their transmissions. If the geometry of the fingertip is known, one can use *intrinsic tactile sensing* [19.50, 94] to compute the contact location as well as the contact force by examining ratios of resultant forces and torques at the sensor.

When the contacts are small compared to the fingertips, so that a point-contact approximation applies, and the fingertips are convex shapes, the contact location is easily computed. Figure 19.7 shows a contact force, \mathbf{f} , contacting the fingertip surface at a location \mathbf{r} . A force/torque sensor such as that in Fig. 19.2 measures the moment, $\boldsymbol{\tau} = \mathbf{r} \times \mathbf{f}$ with respect to the origin. If we consider the lever arm, \mathbf{h} , perpendicular to the line of action of \mathbf{f} , then $\mathbf{h}/h = \mathbf{f}/f \times \boldsymbol{\tau}/\tau$, where $h = \tau/f$ is the magnitude of \mathbf{h} . We can then write that $\mathbf{r} = \mathbf{h} - \alpha \mathbf{f}$, where α is a constant obtained by solving for the intersection of the line of action and the fingertip surface. For a convex fingertip, there will be two such points, of which only one corresponds to a positive (inward) contact force.

Going a couple of steps further, from the contact location one can deduce the local contact normal and contact kinematic type from a small number of force

measurements. Bicchi presents algorithms for extending these methods to soft fingers [19.94]. Brock and Chiu [19.51] describe the use of force sensors for the perception of object shape using this approach, and for measuring the mass and center of mass of a grasped object.

For precision tasks involving small objects or small forces and motions, cutaneous sensors provide the most sensitive measurements. In general, as task requirements get smaller, the sensor must be located closer to the contact so that the compliance and inertia of the intervening parts of the manipulator do not interfere with the measurement. Dario [19.95] suggests that fingertip force sensors are useful for forces of 0.1–10.0 N while array sensors can measure distributed forces of 0.01–1.0 N. Son et al. [19.11] find that intrinsic tactile sensing and array sensors can both provide accurate (within 1 mm) estimates of contact location; however the intrinsic tactile sensing method is inherently sensitive to the accuracy of the force/torque sensor calibration and can produce transient errors due to unmodeled dynamic forces.

Proceeding down the left side of Fig. 19.6 we come to the large category of cutaneous array sensors. The interpretation of information from array sensors depends initially on the transducer type. For arrays of binary contact or proximity sensors interpretation amounts mainly to establishing the location and general shape of the contact area. Techniques common to binary vision can be used to obtain subpixel resolution and to identify contact features. This information, in combination with measurements of the grasp forces from actuator effort or force/torque sensors, is sufficient for basic manipulation tasks [19.47].

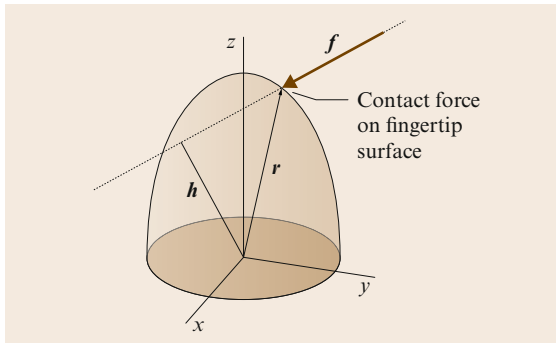


Fig. 19.7 Intrinsic tactile sensing: a contact produces a unique line of action and moment about the origin of a coordinate system in the fingertip. The contact location can be obtained by solving for the intersection of the line of action and the fingertip surface

19.2.2 Solid Mechanics and Deconvolution

A basic problem associated with tactile array sensors is to reconstruct what is happening at the surface of the skin from a finite set of measurements obtained beneath the surface. Typically we are interested in determining pressure and perhaps shear stress distributions associated with contacts on the skin. In other cases, as when the fingertips consist of a gel or soft foam covered by a thin membrane so that the pressure is nearly constant, the local geometry of the contact is of interest.

In the following example we consider the case of an array of elements located at a depth d below the surface of an elastomeric skin. A contact has resulted in a pressure distribution over the region of interest. We establish a coordinate system with z pointing in the inward normal direction and, for simplicity, we examine a one-

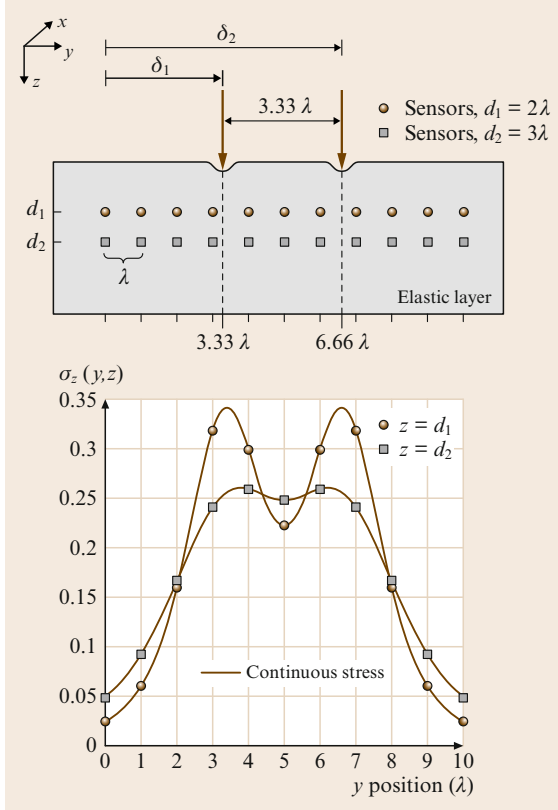


Fig. 19.8 Plane-strain stress response for two (unit-magnitude) line loads. Note the *blurring* that occurs with greater depth

dimensional loading case, $p(y)$, in which the pressure distribution is unchanging in the x -direction. We further assume that the extent of the skin in the x -direction is large compared to the skin thickness so that strains in the x -direction are inhibited, leading to a plane-strain elasticity problem and we assume that the skin is a homogeneous, isotropic material and that the strains are small enough that linear elasticity theory can be applied. Of course, none of these assumptions is entirely valid in practical cases; however, the results do agree qualitatively with the measurements obtained with actual robot fingers and tactile arrays. A thorough discussion of the general approach and of the accuracy of the linear elastic models can be found in [19.22, 79, 96–98].

Figure 19.8 illustrates the case of two line loads, or knife edges, pressed against the surface of the skin (akin to a planar version of the two-point discrimination test for human tactile acuity). The solution for a single line load, or impulse response, was derived by Boussinesq in

1885. For the case of plane strain the principal stresses in the (y, z) -plane from a normal unit impulse can be expressed in Cartesian coordinates as [19.99]:

$$\sigma_z(y, z) = \left(\frac{-2}{\pi z} \right) \frac{1}{[1 + (y/z)^2]^2}, \quad (19.1)$$

$$\sigma_y(y, z) = \left(\frac{-2}{\pi z} \right) \frac{(y/z)^2}{[1 + (y/z)^2]^2}, \quad (19.2)$$

$$\sigma_x(y, z) = \nu(\sigma_y + \sigma_z), \quad (19.3)$$

where ν is Poisson's ratio for the material (typically 0.5 for elastomeric materials).

For two such line loads located at distances δ_1 and δ_2 from the origin, the solution can be obtained by superposition:

$$\begin{aligned} \sigma_z(y, z) &= \left(\frac{-2}{\pi z} \right) \left(\frac{1}{[1 + (\frac{y-\delta_1}{z})^2]^2} + \frac{1}{[1 + (\frac{y-\delta_2}{z})^2]^2} \right), \end{aligned} \quad (19.4)$$

$$\begin{aligned} \sigma_y(y, z) &= \left(\frac{-2}{\pi z} \right) \left(\frac{(\frac{y-\delta_1}{z})^2}{[1 + (\frac{y-\delta_1}{z})^2]^2} + \frac{(\frac{y-\delta_2}{z})^2}{[1 + (\frac{y-\delta_2}{z})^2]^2} \right). \end{aligned} \quad (19.5)$$

For more general pressure distributions the stresses can be found by convolution of the pressure distribution $p(y)$ and the impulse response $G_i(y, z)$:

$$\sigma_i = \int_{\tau=-\infty}^{\tau=y} [p(\tau) d\tau] G_i(y - \tau, z). \quad (19.6)$$

Also plotted in Fig. 19.8 are curves corresponding to the vertical stress components, σ_z , at two different depths, $d_1 = 2\lambda$ and $d_2 = 3\lambda$, where λ is the sensor spacing. As we go deeper beneath the skin, the stresses become smoothed or blurred, and the ability to distinguish between closely spaced impulses diminishes. However, the blurring of concentrated pressure distributions can also provide an advantage when we have a limited number of sensors because the stresses and strains spread over a larger area and are more likely to affect at least one sensor. The elastic skin also provides a kind of automatic *edge enhancement* because stresses are high at the

transitions between loaded and unloaded regions of the skin.

In most cases, for example, in the case of capacitive or magnetic sensors, the sensing elements will measure strains or local deformations of the skin material in the vertical direction. In a few cases, such as pieces of piezoelectric film embedded in an elastomeric skin [19.22], the sensors are sufficiently stiff compared to the surrounding material that they can be considered to measure stresses directly.

For the case of elastic plane strain, the strains are related to the stresses by [19.100]:

$$\epsilon_y = \frac{1}{E} [\sigma_y - \nu(\sigma_x + \sigma_z)] , \quad (19.7)$$

$$\epsilon_z = \frac{1}{E} [\sigma_z - \nu(\sigma_x + \sigma_y)] , \quad (19.8)$$

where E is the Young's modulus and ν is the Poisson's ratio, which we assume is 0.5 for an elastomeric skin.

Figure 19.9 shows the typical measurements that might be obtained from a row of sensing elements from the two line loads applied in Fig. 19.8. Each bar corresponds to the strain, ϵ_{zi} measured by a corresponding element and computed using (19.8), with stresses obtained from (19.4), (19.5), and (19.3).

The problem at this point is to produce a best estimate of the surface pressure distribution, $p(y)$, from this finite set of subsurface strain measurements. The problem is a classic example of estimating a signal from a sparse set of remote measurements. One approach to this process is based on deconvolution techniques [19.22, 79, 97]. The measured signal from the sensors ϵ_z is convolved with the inverse of the impulse strain response $H(y)$ to find

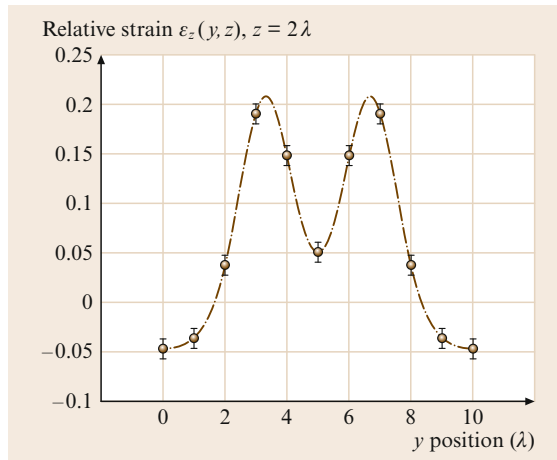


Fig. 19.9 Measured strain with assumed 5% noise

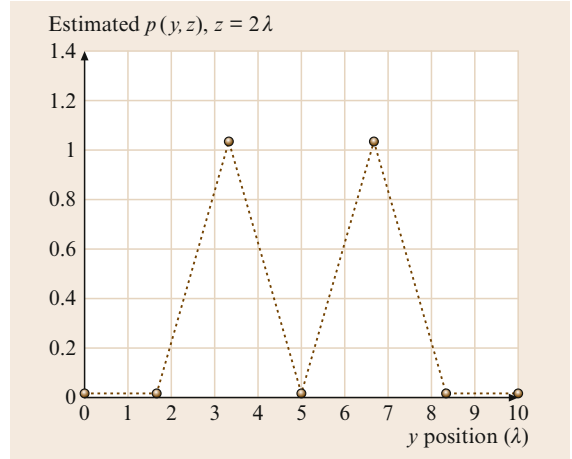


Fig. 19.10 Estimated surface pressure distribution using pseudoinverse method for 11 sensors and seven assumed impulses

an estimate of the surface pressure that produced the signals. The inversion tends to amplify high-frequency noise and the inverse filter bandwidth must be limited according to the spatial density of the sensors and their depth below surface.

Another approach [19.16, 96] is to assume that the surface pressure distribution can be approximated by a finite set of impulses $\mathbf{p} = [p_1, p_2, \dots, p_n]^t$. The sensor readings form a vector, $\boldsymbol{\epsilon} = [\epsilon_1, \epsilon_2, \dots, \epsilon_m]^t$, where $m > n$ for the bandwidth limitations discussed above. The strain response can then be written as a matrix equation

$$\boldsymbol{\epsilon} = H\mathbf{p} . \quad (19.9)$$

Each element of H is computed using (19.8) with σ_z and σ_y computed using equations similar to (19.4) and (19.5), and with σ_x from (19.3). The estimated discrete pressure distribution is then found by taking the pseudoinverse of H :

$$\hat{\mathbf{p}} = H^+ \boldsymbol{\epsilon} . \quad (19.10)$$

Using the strain measurements from Fig. 19.9, at a depth $d = 2\lambda$, the estimated pressure distribution using the pseudoinverse method is seen in Fig. 19.10. In this example, because the assumed set of seven impulses matches fortuitously with the actual loading, the reconstruction is fairly accurate despite the assumed 5% noise.

An alternative approach to constructing soft robot finger tips is to enclose a compliant medium such as

foam rubber or fluid in a thin elastic membrane [19.18, 80, 84, 101–104]. Some of the tactile array sensors developed for these fingers are able to measure directly the shape of the membrane, so that a physical model is not needed for interpretation of the signal [19.18]. Another sensing scheme uses an array of magnetic sensors at the center of the finger to measure the changes in the magnetic field generated by deformations of the magnet-loaded membrane [19.104]. A statistical algorithm that can robustly determine the membrane shape from the sensor signals has been developed [19.18]. However, a mechanical model is still required to find the pressure distribution across the contact from the shape information provided by all of these sensors.

19.2.3 Curvature and Shape Information

Another alternative to measuring subsurface strains or deflections is to measure directly the local curvature at each element of an array of sensors [19.105, 106]. The curvature information can be applied directly toward identifying contact type and centroid location or it can be integrated to obtain the local shape of the contact as with sensors just described that measure the profile of a membrane. To reduce the effects of noise it is useful to assume a simple model for the membrane and use basis functions to fit the curvature data before integrating [19.20].

Returning to Fig. 19.6, once the local contact shape or geometry has been established the next steps typically include feature identification (e.g., identifying corners or ridges on an object) and determining the overall shape and pose of the object in the hand.

Often the object shape is at least partially known a priori, in which case a variety of surface or data fitting methods can be used, for example, *Fearing* [19.107] developed a method for calculation of the radius of curvature and orientation of generalized cylinders from tactile array data and [19.108] developed a neural network that performs a similar calculation. Other schemes use contact locations, surface normals, and contact forces to determine information about object shape and orientation with respect to the hand [19.109–112].

Allen [19.113] uses several different primitive representations for object shape attributes based on the particular exploratory procedure used to sense the object. Object volume and approximate shape are perceived with enclosure grasping, and the resulting shape is modeled using superquadric surfaces. Similarly, measurement of the lateral extent of object faces leads to

a face–edge–vertex model and contour following to a generalized cylinder representation.

The question of what constitutes an appropriate set of features is not well understood, although it clearly depends upon the intended application. *Ellis* [19.69] considers appropriate feature sets and methodologies for acquiring the needed data. *Lederman* and *Browse* [19.114] suggest that surface roughness, surface curvature, and oriented edges are used in human haptic perception.

19.2.4 Object and Surface Identification

The most common application of touch information has been in object recognition and classification. In object recognition the goal is to identify one object from a set of known objects using information derived from touch. In classification the goal is to categorize objects according to preselected sensed properties. These systems are usually based on geometric information derived from tactile array or force sensors. Recently the use of other types of touch information (e.g., compliance, texture, thermal properties) in exploration and identification has received some attention [19.115–117].

A number of systems have used statistical pattern recognition, which can improve noise immunity since only statistics derived from the sensed data are used. Unfortunately, this also means that only a few object types can be discriminated. Systems have been based on the statistics of tactile array sensor data [19.118, 119], and on the statistics of finger joint angles while grasping the object [19.120–122].

Many different features derived from tactile array data have been used for model-based recognition and classification. Systems have used geometric features such as holes, edges, and corners [19.105, 114, 123] and object surfaces [19.124]. Other feature sets include geometric moments [19.125, 126], linear transforms [19.127], and sequences of surface tangents [19.128]. *Gaston* and *Lozano-Perez* [19.110] use local surface normals and contact locations as features, which could be derived from array, force, or joint sensor information. *Siegel* [19.129] presents a method for finding the pose of a known object grasped in a robot hand from measurements of the finger joint angles and torques.

19.2.5 Active Sensing Strategies

Because touch provides only local information, movement is an integral part of touch sensing for recognition

and exploration. Several workers in object recognition applications have developed strategies for scheduling sensor movements so that each additional observation decreases the number of objects which are consistent with prior observations. This is sometimes described as a *hypothesize and test* approach. Early examples include Gurfinkel et al. [19.105] and Hillis [19.123]. Schneider [19.111], Grimson and Lozano-Perez [19.109], and Ellis [19.130] and Cameron [19.131] have developed algorithms for scheduling sensor moves to rapidly recognize polygonal objects using touch measurements of contact location and local surface normal. In Schneider's scheme, each sensing move should cross the boundary of the intersection of all objects consistent with previous sensor observations. Yap and Cole [19.112] show that a probing strategy to determine the shape of a convex planar polygon with V vertices requires at least $3V - 1$ measurements. Roberts [19.132] proposes a movement strategy for recognition that includes tracing the robot finger along object surfaces and edges, rather than moving the finger through free space between readings.

In non-model-based approaches, Klatzky et al. [19.117] have suggested that robotics systems can employ the same exploratory procedures used by humans in haptic exploration. These procedures prescribe the finger motions needed for tasks such as tracing object contours, measuring compliance, and determining the lateral extent of object surfaces. Stansfield [19.116, 133] and Allen [19.113] have implemented some of these exploratory procedures using multifingered robot hands with tactile array sensors. Dario et al. [19.134] have created similar *tactile subroutines*, which they have used in a number of interesting applications, including the detection of hardened lumps (e.g., tumors) beneath a compliant surface. Kaneko and Tanie [19.45] describe a method for using small finger motions to find the location of finger-object contacts without distributed tactile sensors.

Edge tracking and surface following have also received considerable attention. Muthukrishnan et al. [19.135] look at edge finding algorithms for matching segments between successive tactile array impressions. Berger and Khosla [19.136] have demonstrated curved edge tracking in real time using tactile array information. Pribadi et al. [19.137] propose a control strategy for tracking an unknown object surface using tactile information, and Bay [19.138] designs a surface shape estimator for exploration by a multi-

fingered hand which uses contact location and surface normal information from a force sensor. Zhang and Chen [19.139] present an approach to *tactile servoing* in which they model the contact between a compliant tactile sensor and an object to obtain a tactile Jacobian, which is used to produce incremental motion commands for a manipulator. They demonstrate the approach for rolling contact and edge tracking tasks.

In practice, surface following and manipulation are often combined. Okamura and Cutkosky [19.140] present an approach in which a rounded fingertip equipped with a tactile sensor is used to trace object surfaces and locate features, which are defined as regions having local curvature that is high in comparison to that of the fingertip.

19.2.6 Dynamic Sensing and Event Detection

For dynamic tactile sensors used to detect such events as gentle contacts or slippage between the fingertips and an object, the main challenge is to detect the event in question reliably, without false positives. The dynamic tactile sensors that produce large signals in response to contact events are also prone to producing large signals in response to vibrations from the robot drive train and to rapid accelerations of the robot hand. Solutions for more robustly detecting contact events include comparing the signals from dynamic tactile sensors at and away from the contact regions and statistical pattern recognition methods to identify the *signature* of true contact events [19.65, 68, 93].

19.2.7 Integration of Thermal and Other Sensors

Sensors such as thermal contact sensors are rarely used in isolation; their signals are generally integrated with those from tactile arrays and other sensors to produce additional information for identifying objects. For example, Dario et al. [19.141] demonstrate an approach in which a thermal contact sensor is used in combination with a tactile pressure sensing array and a dynamic tactile sensor used to characterize surface roughness, for discriminating among different objects. Different exploratory procedures, inspired by those used by humans [19.142] are called upon to resolve ambiguities.

19.3 Integration Challenges

A critical problem that we have not yet addressed is the difficulty of connecting to a large and diverse array of tactile sensors. In 1987 *Jacobsen et al.* [19.143] cited the routing of wires as perhaps the most difficult problem in dexterous hand design and, to a large extent, this remains true today. However, some solutions to this problem have been presented in recent years by either using wireless sensors or by use of clever bussing for power and signal connections. *Shinoda and Oasa* [19.144] embed tiny wireless sensing elements in an elastic skin that uses an inductive base coil to provide power and signal transmission. Each sensing element is a tuned resonator with a distinct resonant frequency whose resonant frequency is stress sensitive.

Yamada et al. [19.68] use wireless sensor chips that use light transmitted through a transparent elastomer both for power and to communicate six stress components to a power-receiver chip positioned safely beneath the skin's surface. *Hakozaki and Shinoda* [19.145] embed tactile sensor chips between two layers of conductive rubber that serve to provide means for power as well as a bussed serial communication, thus eliminating discrete wiring. The shrinking dimensions of micro-processors also make it possible to mount devices for multiplexing, signal conditioning, etc. in the immediate proximity of the sensors, thereby reducing the amount of *raw* information that must be relayed back to the robot.

19.4 Conclusions and Future Developments

In comparison to computer vision, tactile sensing always seems to be *a few years away* from widespread adoption. As explained in the introduction to this chapter, the reasons include physical problems (placement and robustness of sensors, wiring challenges) and the diversity of sensor types for detecting forces, pressures, local geometries, vibrations, etc. As we have seen, the transduction and interpretation methods are typically different for each of these tactile quantities. However, there are some basic issues that apply to tactile sensing in general; for example, sensors are generally located within or beneath a compliant skin, which affects the quantities that they sense in comparison to pressures, stresses, thermal gradients or displacements applied to the skin surface.

When choosing tactile sensors for a robot arm or hand, it is effective to begin with a consideration of which tactile quantities are most desired and for what purpose; for example, the main concern is to obtain accurate measurements of loads or contact forces at sufficient

data rates for force servoing, then intrinsic tactile sensing may make the most sense. If manipulating objects with soft contacts and with sliding or rolling, curved array sensors for measuring pressure distributions, or perhaps local skin deflections, may be desirable. If exploring objects to learn about their texture and material composition, dynamic tactile sensors and thermal sensors may be effective.

In an ideal world, one would incorporate all these tactile sensors in a robotic end-effector without regard to cost, signal processing or wiring complexity. Fortunately, the cost and size of transducers suitable for tactile sensing are steadily dropping and the ability to perform localized processing is improving with surface-mounted devices on flexible circuits. In the near future it will be increasingly possible to fabricate dense arrays of transducers in situ on contoured surfaces, using material deposition and laser machining techniques. In this way, robots may finally start to approach the tactile sensitivity and responsiveness of the simplest of animals.

References

- 19.1 M.H. Lee: Tactile sensing: new directions, new challenges, *Int. J. Robot. Res.* **19**(7), 636–643 (2000)
- 19.2 M.H. Lee, H.R. Nicholls: Tactile sensing for mechatronics – a state of the art survey, *Mechatronics* **9**(1), 1–31 (1999)
- 19.3 J.-P. Uldry, R.A. Russell: Developing conductive elastomers for applications in robotic tactile sensing, *Adv. Robot.* **6**(2), 255–271 (1992)
- 19.4 T.V. Papakostas, J. Lima, M. Lowe: A large area force sensor for smart skin applications, *Proc. IEEE Sensors*, Vol. 2 (2002) pp. 1620–1624
- 19.5 T. Someya: Integration of organic field-effect transistors and rubbery pressure sensors for artificial skin applications, *IEEE Int. Electron. Dev. Meeting* (Washington 2003) pp. 8–14

- 19.6 M. Shimojo, A. Namiki, M. Ishikawa, R. Makino, K. Mabuchi: A tactile sensor sheet using pressure conductive rubber with electrical-wires stitched method, *IEEE Sens. J.* **4**(5), 589–596 (2004)
- 19.7 D. De Rossi, A. Della Santa, A. Mazzoldi: Dressware: wearable piezo- and thermoresistive fabrics for ergonomics and rehabilitation, *Proc. 19th Ann. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Vol. 5 (Elsevier 1997) pp. 1880–1883
- 19.8 A. Tognetti, F. Lorusi, M. Tesconi, D. De Rossi: Strain sensing fabric characterization, *Proc. IEEE Sensors*, Vol. 1 (2004) pp. 527–530
- 19.9 R.S. Fearing, T.O. Binford: Using a cylindrical tactile sensor for determining curvature, *IEEE Trans. Robot. Autom.* **7**(6), 806–817 (1991)
- 19.10 D.C. Chang: Tactile Array Sensors and Data Interpretation for Dexterous Robotic Rolling Manipulation. Ph.D. Thesis (Stanford University, Stanford 1995)
- 19.11 J.S. Son, M.R. Cutkosky, R.D. Howe: Comparison of contact sensor localization abilities during manipulation, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Vol. 2 (1995) pp. 96–103
- 19.12 Pressure Profiles Systems: www.pressureprofile.com/
- 19.13 T. Hoshi, H. Shinoda: A sensitive skin based on touch-area-evaluating tactile elements, *Proc. IEEE Virtual Reality* (2006) pp. 89–94
- 19.14 B.J. Kane, M.R. Cutkosky, G.T.A. Kovacs: A traction stress sensor array for use in high-resolution robotic tactile imaging, *J. Microelectromech. Syst.* **9**(4), 425–434 (2000)
- 19.15 P. Valdastri, S. Roccella, L. Beccai, E. Cattin, A. Men- ciassi, M.C. Carrozza, P. Dario: Characterization of a novel hybrid silicon three-axial force sensor, *Sens. Actuators A* **123–124**, 249–257 (2005)
- 19.16 K. Kamiyama, H. Kajimoto, N. Kawakami, S. Tachi: Evaluation of a vision-based tactile sensor, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 2 (2004) pp. 1542–1547
- 19.17 N.J. Ferrier, R.W. Brockett: Reconstructing the shape of a deformable membrane from image data, *Int. J. Robot. Res.* **19**(9), 795–816 (2000)
- 19.18 W. Nowlin: Experimental results on Bayesian algorithms for interpreting compliant tactile sensor data, *Proc. IEEE Int. Conf. Robot. Autom.* (Sacramento 1991) pp. 378–383
- 19.19 R.A. Russell, S. Parkinson: Sensing surface shape by touch, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 1 (1993) pp. 423–428
- 19.20 W.R. Provancher, M.R. Cutkosky: Sensing local geometry for dexterous manipulation, *Proc. Int. Symp. Exp. Robot.* (Sant'Angelo d'Ischia 2002) pp. 507–516
- 19.21 P. Dario, R. Lazzarini, R. Magni, S.R. Oh: An integrated miniature fingertip sensor, *Micro Machine and Human Science*, 1996., *Proc. 7th International Symposium* (1996) pp. 91–97
- 19.22 R.D. Howe, M.R. Cutkosky: Dynamic tactile sensing: Perception of fine surface features with stress rate sensing, *IEEE Trans. Robot. Autom.* **9**(2), 140–151 (1993)
- 19.23 H. Gladstone, K.J. Waldron, C. Enedah: Stiffness and texture perception for teledermatology, *Stud. Health Technol. Inform.* **111**, 579–585 (2005)
- 19.24 R.D. Howe, M.R. Cutkosky: Sensing skin acceleration for texture and slip perception, *Proc. IEEE Int. Conf. Robot. Autom.* (Scottsdale 1989) pp. 145–150
- 19.25 R.A. Russell: Using tactile whiskers to measure surface contours, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 2 (1992) pp. 1295–1299
- 19.26 M. Kaneko, N. Ueno, T. Tsuji: Active antenna-basic considerations on the working principle, *Proc. IEEE/RSJ/GI Int. Conf. Intell. Robot. Syst. (IROS '94)*, Vol. 3 (1994) pp. 1744–1750
- 19.27 T.N. Clements, C.D. Rahn: Three-dimensional contact imaging with an actuated whisker, *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS 2005)* (2005) pp. 598–603
- 19.28 N. Cowan, E. Ma, M.R. Cutkosky, R. Full: A biologically inspired passive antenna for steering control of a running robot, *International Symposium on Robotics Research*, Springer Tracts Adv. Robotics (Springer, Berlin, Heidelberg 2004)
- 19.29 J.M. Vranish, R.L. McConnell, S. Mahalingam: Capaciflector collision avoidance sensors for robots, *Comput. Electron. Eng.* **17**(3), 173–179 (1991)
- 19.30 E. Cheung, V. Lumelsky: A sensitive skin system for motion control of robot arm manipulators, *Robot. Autom. Syst.* **10**(1), 9–32 (1992)
- 19.31 D. Um, B. Stankovic, K. Giles, T. Hammond, V. Lumelsky: A modularized sensitive skin for motion planning in uncertain environments, *Proc. 1998 IEEE Int. Conf. Robot. Autom.*, Vol. 1 (Leuven 1998) pp. 7–12
- 19.32 S. Walker, K. Loewke, M. Fischer, C. Liu, J.K. Salisbury: An optical fiber proximity sensor for haptic exploration, *Int. Conf. Robot. Autom.* (IEEE 2007) pp. 473–478
- 19.33 E. Guglielmelli, V. Genovese, P. Dario, G. Morana: Avoiding obstacles by using a proximity US/IR sensitive skin, *Proc. 1993 IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS '93)* (Yokohama 1993) pp. 2207–2214
- 19.34 D. Wegerif, D. Rosinski: Sensor based whole arm obstacle avoidance for kinematically redundant robots, *Proc. SPIE Int. Soc. Opt. Eng.*, Vol. 1828 (Boston 1992) pp. 417–426
- 19.35 G. Buttazzo, P. Dario, R. Bajcsy: Finger based explorations. In: *Intelligent Robots and Computer Vision: Fifth in a Series*, ed. by D. Casasent (Cambridge 1986) pp. 338–345
- 19.36 D. Siegel, I. Garabieta, J. Hollerbach: An integrated tactile and thermal sensor, *Proc. IEEE Int. Conf. Robot. Autom.* (San Francisco 1986) pp. 1286–1291
- 19.37 R.A. Russell: A thermal sensor array to provide tactile feedback for robots, *Int. J. Robot. Res.* **5**(3), 35–39 (1985)

- 19.38 D.G. Caldwell, C. Gosney: Enhanced tactile feed-back (Tele-taction) using a multi-functional sensory system, Proc. IEEE Int. Conf. Robot. Autom., Vol.1 (Atlanta 1993) pp. 955–960
- 19.39 G.J. Monkman, P.M. Taylor: Thermal tactile sensing, IEEE Trans. Robot. Autom. **9**(3), 313–318 (1993)
- 19.40 J. Engel, J. Chen, X. Wang, Z. Fan, C. Liu, D. Jones: Technology development of integrated multi-modal and flexible tactile skin for robotics applications, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS 2003), Vol. 3 (2003) pp. 2359–2364
- 19.41 P. Bergveld: Development and application of chemical sensors in liquids. In: *Sensors and Sensory Systems for Advanced Robotics* (Springer, Berlin, Heidelberg 1986) pp. 397–416
- 19.42 T. Nakamoto, A. Fukuda, T. Moriizumi: Perfume and flavor identification by odor sensing system using quartz-resonator sensor array and neural-network pattern recognition, TRANSDUCERS '91: Proc. 6th Int. Conf. Solid-State Sensors and Actuators (San Francisco 1991) pp. 355–358
- 19.43 B.A. Auld, A.J. Bahr: A novel multifunction robot sensor, Proc. IEEE Int. Conf. Robot. Autom. (San Francisco 1986) pp. 1262–1267
- 19.44 H. Clergeot, D. Placko, J.M. Detriche: Electrical proximity sensors. In: *Sensors and Sensory Systems for Advanced Robotics*, ed. by P. Dario (Springer, Berlin, Heidelberg 1986) pp. 295–308
- 19.45 M. Kaneko, K. Tanie: Contact point detection for grasping of an unknown object using self-posture changeability (SPC), Proc. IEEE Int. Conf. Robot. Autom. (Cincinnati 1990) pp. 864–869
- 19.46 J.K. Salisbury: Kinematic and force analysis of articulated hands. In: *Robot Hands and the Mechanics of Manipulation*, ed. by M.T. Mason, J.K. Salisbury (MIT Press, Cambridge 1985)
- 19.47 B.B. Edin, L. Beccai, L. Ascari, S. Roccella, J.J. Cabibihan, M.C. Carrozza: A bio-inspired approach for the design and characterization of a tactile sensory system for a cybernetic prosthetic hand, Proc. IEEE Int. Conf. Robot. Autom. (2006) pp. 1354–1358
- 19.48 A. Pugh (Ed.): *Robot sensors, Volume 2: Tactile and Non-Vision* (IFS Springer, New York 1986)
- 19.49 J.G.. Webster: *Tactile Sensors for Robotics and Medicine*, Vol. 16 (Wiley, New York 1988)
- 19.50 J.K. Salisbury: Interpretation of contact geometries from force measurements. In: *Robotics Research: The First International Symposium*, ed. by M. Brady, R.P. Paul (MIT Press, Cambridge 1984)
- 19.51 D. Brock, S. Chiu: Environment perception of an articulated robot hand using contact sensors, ASME Winter Annual Meeting, Robotics and Manufacturing Automation, Vol. 15 (Miami 1985) pp. 89–96
- 19.52 F.W. Sinden, R.A. Boie: A planar capacitive force sensor with six degrees of freedom, Proc. IEEE Int. Conf. Robot. Autom. (San Francisco 1986) pp. 1806–1813
- 19.53 Y.-L. Park, K. Chau, R.J. Black, M.R. Cutkosky: Force sensing robot fingers using embedded fiber Bragg grating sensors and shape deposition manufacturing, Int. Conf. Robot. Autom. (IEEE 2007) pp. 1510–1516
- 19.54 A. Bicchi: A criterion for optimal design of multiaxis force sensors, MIT AI Lab Memo, no. 1263 (MIT Press, Cambridge 1990)
- 19.55 M. Uchiyama, E. Bayo, E. Palma-Villalon: A mathematical approach to the optimal structural design of a robot force sensor, Proc. USA–Japan Symp. Flexible Automation (ASME, Minneapolis 1988) pp. 539–546
- 19.56 A. Bicchi, J.K. Salisbury, P. Dario: Augmentation of grasp robustness using intrinsic tactile sensing, IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 302–307
- 19.57 M. Ueda: Tactile sensors for an industrial robot to detect a slip, Proc. 2nd Int. Symp. Industrial Robots (Chicago 1972) pp. 63–70
- 19.58 R. Matsuda: Slip sensor of industrial robot and its application, Electric. Eng. Jap. **96**(5), 129–136 (1976)
- 19.59 J. Rebman, J.-E. Kallhammer: A search for precursors of slip in robotic grasp. In: *Intelligent Robots and Computer Vision: Fifth in a Series*, ed. by D. Casasent (Cambridge 1986) pp. 329–337
- 19.60 P. Dario, D. De Rossi: Tactile sensors and the gripping challenge, IEEE Spectrum **22**(5), 46–52 (1985)
- 19.61 M.R. Cutkosky, R.D. Howe: Dynamic tactile sensing. In: *ROMANSY 88: Seventh CISM-IFTOMM Symposium on the Theory and Practice of Robots and Manipulators*, ed. by A. Morecki, G. Bianchi, K. Jaworek (Udine 1988)
- 19.62 D. Dornfeld, C. Handy: Slip detection using acoustic emission signal analysis, Proc. IEEE Int. Conf. Robot. Autom., Vol. 3 (1987) pp. 1868–1875
- 19.63 J.F. Cuttine, C.O. Huey, T.D. Taylor: Tactile sensing of incipient slip, Proc. USA–Japan Symp. Flexible Automation (ASME, Minneapolis 1988) pp. 547–555
- 19.64 J.B. Morrell: Force Modulation of a Robot Gripper Using Slip Detection. Ph.D. Thesis (Univ. of Washington, Washington 1990)
- 19.65 M.R. Tremblay, M.R. Cutkosky: Estimating friction using incipient slip sensing during a manipulation task, Proc. IEEE Int. Conf. Robot. Autom. (Atlanta 1993) pp. 429–434
- 19.66 J.S. Son, E.A. Monteverde, R.D. Howe: A tactile sensor for localizing transient events in manipulation, Proc. IEEE Int. Conf. Robot. Autom., Vol.1 (1994) pp. 471–476
- 19.67 R.W. Patterson, G.E. Nevill Jr: The induced vibration touch sensor – A new dynamic touch sensing concept, Robotica **4**, 27–31 (1986)
- 19.68 K. Yamada, K. Goto, Y. Nakajima, N. Koshida, H. Shinoda: A sensor skin using wire-free tactile sensing elements based on optical connection, Proc. 41st SICE Ann. Conf., Vol.1 (2002) pp. 131–134

- 19.69 R.E. Ellis: Extraction of tactile features by passive and active sensing. In: *Intelligent Robots and Computer Vision*, ed. by D.P. Casasent (Cambridge 1984)
- 19.70 S. Omata, Y. Terunuma: Development of a new type of tactile sensor for detecting hardness and/or softness of an object like the human hand, TRANS-DUCERS '91: Proc. 6th Int. Conf. Solid-State Sensors and Actuators (IEEE Electron Devices Society, San Francisco 1991) pp. 868–871
- 19.71 A.R. Grahn, L. Astle: Robotic ultrasonic force sensor arrays. In: *Robot sensors, Volume 2: Tactile and Non-Vision*, ed. by A. Pugh (IFS Springer, Berlin, Heidelberg 1986) pp. 297–315
- 19.72 W.C. Nowlin: Experimental results on Bayesian algorithms for interpreting compliant tactile sensing data, Proc. IEEE Int. Conf. Robot. Autom., Vol. 1 (1991) pp. 378–383
- 19.73 J.L. Novak: Initial design and analysis of a capacitive sensor for shear and normal force measurement, Proc. IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 137–145
- 19.74 C.T. Yao: A novel three-dimensional microstructure fabrication technique for a triaxial tactile sensor array, Proc. IEEE Micro Robots and Teleoperators Workshop (Hyannis 1987)
- 19.75 F. Arai, N. Iwata, T. Fukuda: Transparent tactile feeling device for touch-screen interface, Robot and Human Interactive Communication, 13th IEEE International Workshop (ROMAN 2004) pp. 527–532
- 19.76 W. Griffin: Shared Control for Dexterous Telemanipulation with Haptic Feedback. Ph.D. Thesis (Stanford University, Stanford 2003)
- 19.77 H. Maekawa, K. Tanie, K. Komoriya, M. Kaneko, C. Horiguchi, T. Sugawara: Development of a finger-shaped tactile sensor and its evaluation by active touch, Proc. IEEE Int. Conf. Robot. Autom., Vol. 2 (1992) pp. 1327–1334
- 19.78 R.S. Fearing: Tactile sensing, perception and shape interpretation. Ph.D. Thesis (Stanford University, Stanford 1987)
- 19.79 R.S. Fearing: Tactile sensing mechanisms, Int. J. Robot. Res. **9**(3), 3–23 (1990)
- 19.80 R. Russell: Compliant-skin tactile sensor, Proc. IEEE Int. Conf. Robot. Autom., Vol. 4 (1987) pp. 1645–1648
- 19.81 O. Kerpa, K. Weiss, H. Worn: Development of a flexible tactile sensor system for a humanoid robot, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS 2003), Vol. 1 (2003) pp. 1–6
- 19.82 R. Kageyama, S. Kagami, M. Inaba, H. Inoue: Development of soft and distributed tactile sensors and the application to a humanoid robot, Proc. IEEE Int. Conf. Syst., Man Cybern., Vol. 2 (SMC, 1999) pp. 981–986
- 19.83 K. Noda, I. Shimoyama: A shear stress sensing for robot hands orthogonal arrayed piezoresistive cantilevers standing in elastic material, 14th Symp. Haptic Interfaces for Virtual Environment and Teleoperator Systems (2006) pp. 63–66
- 19.84 R.W. Brockett: Robotic hands with rheological surfaces, IEEE Int. Conf. Robot. Autom. (IEEE Comput. Soc. Press, St. Louis 1985) pp. 942–946
- 19.85 K.B. Shimoga, A.A. Goldenberg: Soft robotic fingertips. I. A comparison of construction materials, Int. J. Robot. Res. **15**(4), 320–350 (1996)
- 19.86 H. Shinoda, K. Matsumoto, S. Ando: Acoustic resonant tensor cell for tactile sensing, Proc. IEEE Int. Conf. Robot. Autom., Vol. 4 (1997) pp. 3087–3092
- 19.87 H. Shinoda, S. Sasaki, K. Nakamura: Instantaneous evaluation of friction based on ARTC tactile sensor, Proc. IEEE Int. Conf. Robot. Autom., Vol. 3 (2000) pp. 2173–2178
- 19.88 S. Ando, H. Shinoda, A. Yonenaga, J. Terao: Ultrasonic six-axis deformation sensing, IEEE Trans. Ultrasonics Ferroelectrics Frequency Control **48**(4), 1031–1045 (2001)
- 19.89 D.M. Siegel: Contact sensors for dextrous robotic hands, MIT Artificial Intelligence Laboratory Tech. Rep., no. 900 (Cambridge 1986)
- 19.90 F. Castelli: An integrated tactile-thermal robot sensor with capacitive tactile array, IEEE Trans. Ind. Appl. **38**(1), 85–90 (2002)
- 19.91 T. Shimizu, M. Shikida, K. Sato, K. Itoigawa: A new type of tactile sensor detecting contact force and hardness of an object, Technical Digest 15th IEEE Int. Conf. Micro Electro Mechanical Systems (MEMS, Las Vegas 2002) pp. 344–347
- 19.92 N.S. Methil, Y. Shen, D. Zhu, C.A. Pomeroy, R. Mukherjee: Development of supermedia interface for telediagnosics of breast pathology, Proc. IEEE Int. Conf. Robot. Autom. (2006) pp. 3911–3916
- 19.93 B.S. Eberman, J.K. Salisbury: Determination of manipulator contact information from joint torque measurements. In: *Experimental Robotics I, The First International Symposium*, ed. by V. Hayward, O. Khatib (Springer, Montreal 1990)
- 19.94 A. Bicchi: Intrinsic contact sensing for soft fingers, Proc. IEEE Int. Conf. Robot. Autom. (Cincinnati 1990) pp. 968–973
- 19.95 P. Dario: Tactile sensing for robots: Present and future. In: *The Robotics Review 1*, ed. by O. Khatib, J. Craig, T. Lozano-Perez (MIT Press, Cambridge 1989) pp. 133–146
- 19.96 J.R. Phillips, K.O. Johnson: Tactile spatial resolution III: A continuum mechanics model of skin predicting mechanoreceptor responses to bars, edges and gratings, J. Neurophysiol. **46**(6), 1204–1225 (1981)
- 19.97 R.S. Fearing, J.M. Hollerbach: Basic solid mechanics for tactile sensing, Int. J. Robot. Res. **4**(3), 40–54 (1985)
- 19.98 T. Speeter: A tactile sensing system for robotic manipulation, Int. J. Robot. Res. **9**(6), 25–36 (1990)
- 19.99 K.L. Johnson: *Contact Mechanics* (Cambridge Univ. Press, Cambridge 1985)

- 19.100 S. Timoshenko, J.N. Goodier: *Theory of Elasticity* (McGraw-Hill, New York 1951)
- 19.101 G. Kenaly, M. Cutkosky: Electrorheological fluid-based fingers with tactile sensing, Proc. IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 132–136
- 19.102 R.D. Howe: Dynamic Tactile Sensing. Ph.D. Thesis (Stanford University, Stanford 1990)
- 19.103 R.M. Voyles, B.L. Stavnheim, B. Yap: Practical electrorheological fluid-based fingers for robotic applications, IASTED Int. Symp. Robot. Manuf. (Santa Barbara 1989)
- 19.104 J.J. Clark: A magnetic field based compliance matching sensor for high resolution, high compliance tactile sensing, Proc. IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 772–777
- 19.105 V.S. Gurfinkel: Tactile sensitizing of manipulators, Eng. Cybernet. **12**(6), 47–56 (1974)
- 19.106 T.H. Speeter: Analysis and control of robotic manipulation. Ph.D. Thesis (Case Western Reserve University, Cleveland 1987)
- 19.107 R. Fearing: Tactile sensing for shape interpretation. In: *Dextrous Robot Hands*, ed. by S.T. Venkataraman, T. Iberall (Springer, Berlin, Heidelberg 1990) pp. 209–238
- 19.108 A.J. Worth, R.R. Spencer: A neural network for tactile sensing: The Hertzian contact problem, Int. Joint Conf. Neural Networks (1989) pp. 267–274
- 19.109 W.E.L. Grimson, T. Lozano-Perez: Model-based recognition and localization from sparse range or tactile data, Int. J. Robot. Res. **3**(3), 3–35 (1984)
- 19.110 P.C. Gaston, T. Lozano-Perez: Tactile recognition and localization using object models: The case of polyhedra on a plane, IEEE Trans. Pattern Anal. Mach. Intell. (1984) pp. 257–266
- 19.111 J.L. Schneider: An objective sensing strategy for object recognition and localization, Proc. IEEE Int. Conf. Robot. Autom. (San Francisco 1986) pp. 1262–1267
- 19.112 R. Cole, C. Yap: Shape from probing, J. Algorithms **8**(1), 19–38 (1987)
- 19.113 P.K. Allen: Mapping haptic exploratory procedures to multiple shape representations, Proc. IEEE Int. Conf. Robot. Autom. (Cincinnati 1990) pp. 1679–1684
- 19.114 S.J. Lederman, R. Browse: The physiology and psychophysics of touch. In: *Sensors and Sensory Systems for Advanced Robotics*, ed. by P. Dario (Springer, Berlin, Heidelberg 1986) pp. 71–91
- 19.115 R. Bajcsy: What can we learn from one finger experiments?. In: *Robotics Research: The First International Symposium*, ed. by M. Brady, R.P. Paul (MIT Press, Cambridge 1984) pp. 509–528
- 19.116 S.A. Stansfield: A robotic perceptual system utilizing passive vision and active touch, Int. J. Robot. Res. **7**(6), 138–161 (1988)
- 19.117 R.L. Klatzky, R. Bajcsy, S.J. Lederman: Object exploration in one and two fingered robots, Proc. IEEE Int. Conf. Robot. Autom. (1987) pp. 1806–1809
- 19.118 G.I. Kinoshita: Classification of grasped object's shape by an artificial hand with multi-element tactile sensors. In: *Information Control Problems in Manufacturing Technology*, ed. by Y. Oshima (Pergamon, Oxford 1977) pp. 111–118
- 19.119 M. Briot: The utilization of an 'artificial skin' sensor for the identification of solid objects, 9th Int. Symp. Industrial Robots (1979) pp. 529–547
- 19.120 T. Okada, S. Tsuchiya: Object recognition by grasping, Pattern Recogn. **9**(3), 111–119 (1977)
- 19.121 M. Briot, M. Renaud, Z. Stojilkovic: An approach to spatial pattern recognition of solid objects, IEEE Trans. Syst. Man Cybernet. **SMC-8**(9), 690–694 (1978)
- 19.122 V. Marik: Algorithms of the complex tactile information processing, 7th Int. Joint Conf. Artif. Intell. (1981) pp. 773–774
- 19.123 W.D. Hillis: Active touch sensing, Int. J. Robot. Res. **1**(2), 33–44 (1982)
- 19.124 K.J. Overton, T. Williams: Tactile sensation for robots, Int. Joint Conf. Artif. Intell. **2**, 791–795 (1981)
- 19.125 R.C. Luo, W.-H. Tsai: Object recognition using tactile image array sensors, Proc. IEEE Int. Conf. Robot. Autom. (1986) pp. 1249–1253
- 19.126 R.C. Luo, H.-H. Loh: Recognition of similar objects using a tactile image array sensor. In: *Proceedings of the Intelligent Robots and Computer Vision Conference: Sixth in a Series*, ed. by D. Casasent, E. Hall (Cambridge 1987) pp. 156–163
- 19.127 J. Jurczyk, K.A. Loparo: Mathematical transforms and correlation techniques for object recognition using tactile data, IEEE Trans. Robot. Autom. **5**(3), 359–362 (1989)
- 19.128 H. Ozaki, S. Waku, A. Mohri, M. Takata: Pattern recognition of a grasped object by unit-vector distribution, IEEE Trans. Syst. Man Cybernet. **12**(3), 315–324 (1982)
- 19.129 D. Siegel: Finding the pose of an object in the hand, Proc. IEEE Int. Conf. Robot. Autom. (Sacramento 1991) pp. 406–411
- 19.130 R. Ellis: Acquiring tactile data for the recognition of planar objects, Proc. IEEE Int. Conf. Robot. Autom., Vol. 4 (1987) pp. 1799–1805
- 19.131 A. Cameron: Optimal tactile sensor placement, Proc. IEEE Int. Conf. Robot. Autom. (Scottsdale 1989) pp. 308–313
- 19.132 K. Roberts: Robot active touch exploration: Constraints and strategies, Proc. IEEE Int. Conf. Robot. Autom. (Cincinnati 1990) pp. 1679–1684
- 19.133 S.A. Stansfield: Haptic perception with an articulated, Sensate Robot Hand, Sandia National Laboratories Tech. Rep. SAND90–0085 (1990)
- 19.134 P. Dario: Sensing body structures by an advanced robot system, Proc. IEEE Int. Conf. Robot. Autom. (Philadelphia 1988) pp. 1758–1763
- 19.135 C. Muthukrishnan, D. Smith, D. Meyers, J. Rebman, A. Koivo: Edge detection in tactile images, Proc. IEEE Int. Conf. Robot. Autom. (1987) pp. 1500–1505

- 19.136 A.D. Berger, P.K. Khosla: Using tactile data for real-time feedback, *Int. J. Robot. Res.* **10**(2), 88–102 (1991)
- 19.137 K. Pribadi, J.S. Bay, H. Hemami: Exploration and dynamic shape estimation by a robotic probe, *IEEE Trans Syst. Man Cybernet.* **19**(4), 840–846 (1989)
- 19.138 J.S. Bay: Tactile shape sensing via single- and multi-fingered hands, *Proc. IEEE Int. Conf. Robot. Autom.* (Scottsdale 1989) pp. 290–295
- 19.139 H. Zhang, N.N. Chen: Control of contact via tactile sensing, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 16 (2000) pp. 482–495
- 19.140 A.M. Okamura, M.R. Cutkosky: Feature detection for haptic exploration with robotic fingers, *Int. J. Robot. Res.* **20**(12), 925–938 (2001)
- 19.141 P. Dario, P. Ferrante, G. Giacalone, L. Livaldi, B. Allotta, G. Buttazzo, A.M. Sabatini: Planning and executing tactile exploratory procedures, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Vol. 3 (IEEE 1992) pp. 1896–1903
- 19.142 R.L. Klatzky, S.J. Lederman: Intelligent exploration by the human hand. In: *Dextrous Robot Hands*, ed. by S.T. Venkataraman, T. Iberall (Springer, Berlin, Heidelberg 1990)
- 19.143 S.C. Jacobsen, J.E. Wood, D.F. Knutti, K.B. Biggers: The Utah/MIT dextrous hand: Work in progress. In: *First International Conference on Robotics Research*, ed. by M. Brady, R.P. Paul (MIT Press, Cambridge 1984) pp. 601–653
- 19.144 H. Shinoda, H. Oasa: Passive wireless sensing element for sensitive skin, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS 2000)*, Vol. 2 (2000) pp. 1516–1521
- 19.145 M. Hakozaiki, H. Shinoda: Digital tactile sensing elements communicating through conductive skin layers, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA '02)*, Vol. 4 (2002) pp. 3813–3817

Inertial Sensors

20. Inertial Sensors, GPS, and Odometry

Gregory Dudek, Michael Jenkin

This chapter examines how certain properties of the world can be exploited in order for a robot or other device to develop a model of its own motion or pose (position and orientation) relative to an external frame of reference. Although this is a critical problem for many autonomous robotic systems, the problem of establishing and maintaining an orientation or position estimate of a mobile agent has a long history in terrestrial navigation.

20.1 Odometry	477
20.2 Gyroscopic Systems	479
20.2.1 Mechanical Systems	479
20.2.2 Optical Systems	480

20.2.3 MEMS	481
20.2.4 Performance	481
20.2.5 Summary	482
20.3 Accelerometers	482
20.4 IMU Packages	483
20.5 GPS	484
20.5.1 Overview	484
20.5.2 Performance Factors	486
20.5.3 Enhanced GPS	487
20.5.4 GPS Receivers and Communications	488
20.6 GPS-IMU Integration	488
20.7 Further Reading	489
20.8 Currently Available Hardware	489
References	490

20.1 Odometry

The word *odometry* is a contraction of the Greek words *hodos* meaning *travel* or *journey*, and *metron* meaning *measure*. Given its importance to a wide variety of applications from civil engineering to military conquest, the basic concepts that underly odometry have been studied for over 2000 years. Perhaps the earliest reference to odometry appears in the *Ten Books on Architecture* by Vitruvius, in which he describes *a useful invention of the greatest ingenuity, transmitted by our predecessors, which enables us, while sitting in a carriage on the road or sailing by sea, to know how many miles of a journey we have accomplished* [20.1]. In the context of autonomous vehicles, odometry usually refers to the use of data from the actuators (wheels, treads, etc.) to estimate the overall motion of the vehicle. The basic concept [20.2] is to develop a mathematical model of how selected motions of the vehicle's wheels, joints, etc. induce motion of the vehicle itself, and then to integrate these specified motions over time in order to develop a model of the pose of the vehicle as a function of time.

The use of odometry information to estimate the pose of the vehicle as a function of time is known as *dead reckoning*.

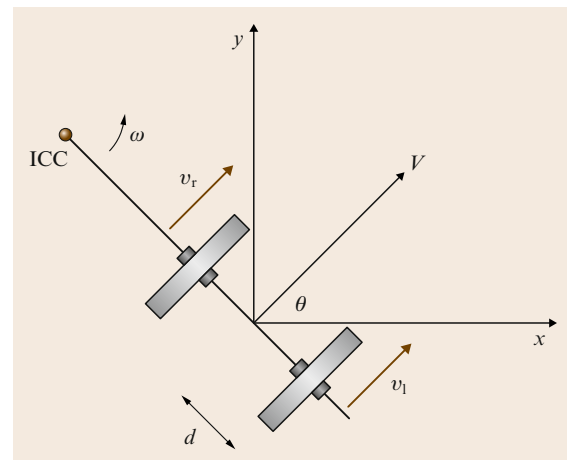


Fig. 20.1 Differential drive kinematics

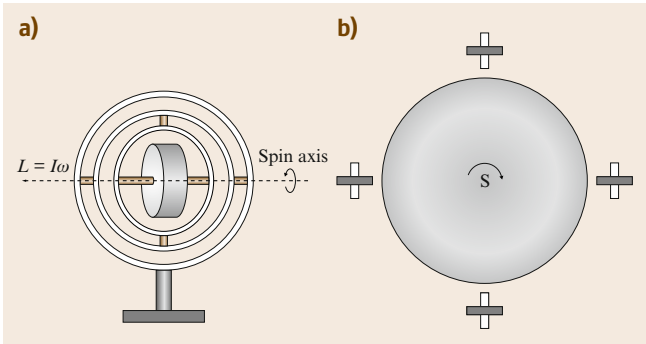


Fig. 20.2a,b Mechanical gyroscope. **(a)** Traditionally gimballed gyroscope. The gimbal provides the gyroscope the freedom to rotate about its axis as the base of the gyroscope is rotated. **(b)** A gyroscope as it is rotated around the planet. The wheel of the gyroscope (grey) remains in the same orientation as it revolves with the planet. To an observer on the planet the gyroscope will appear to rotate

oning or deductive reckoning and finds wide application in navigation at sea [20.3].

The details of odometry estimation varies by vehicle design. In the context of mobile robots perhaps the simplest vehicle for odometry estimation is the differential drive vehicle (Fig. 20.1). A differential drive vehicle has two driveable wheels which are independently controllable and which are mounted along a common axis. Assuming that the location of the wheels are fixed on the vehicle, then for the wheels to remain in constant contact with the ground, the two wheels must describe arcs on the plane such that the vehicle rotates around a point

(known as the ICC – instantaneous center of curvature) that lies on the wheels’ common axis (Fig. 20.1). If the ground contact speeds of the left and right wheels are v_l and v_r respectively, and the wheels are separated by a distance $2d$, then

$$\begin{aligned}\omega(R + d) &= v_l \\ \omega(R - d) &= v_r.\end{aligned}$$

We can rearrange these two equations to solve for ω the rate of rotation about the ICC and R the distance from the center of the robot to the ICC

$$\begin{aligned}\omega &= \frac{(v_l - v_r)}{(2d)}, \\ R &= d \frac{(v_r + v_l)}{(v_l - v_r)}.\end{aligned}$$

The instantaneous velocity of the point midway between the robot’s wheels is given by $V = \omega R$.

Now as v_l and v_r are functions of time we can generate a set of equations of motion for the differential drive robot. Using the point midway between the wheels as the origin of the robot, and writing θ as the orientation of the robot with respect to the x -axis of a global Cartesian coordinate system, one obtains

$$\begin{aligned}x(t) &= \int V(t) \cos(\theta(t)) dt, \\ y(t) &= \int V(t) \sin(\theta(t)) dt, \\ \theta(t) &= \int \omega(t) dt.\end{aligned}$$

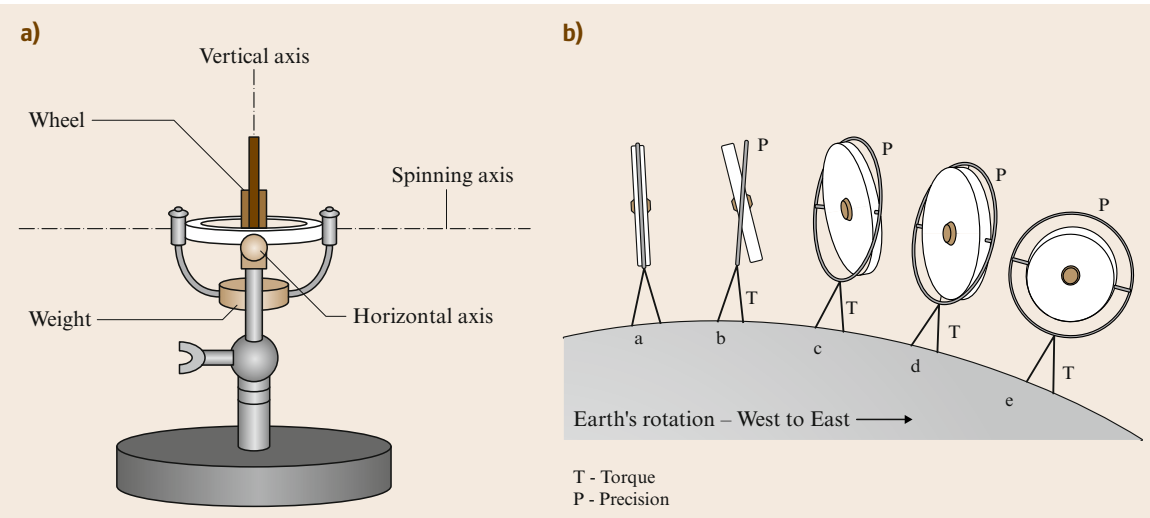


Fig. 20.3a,b Simple gyrocompass. **(a)** Pendulus gyro. **(b)** Precessional motion

This is the solution for the odometry of a differential drive vehicle on the plane. Given the control inputs (v_l and v_r) and some initial state estimate, we can estimate the state of an idealized robot using this motion model at any time t .

Given such a model and complete knowledge of the control inputs, we should, in principle, be able to estimate a robot's pose at any time. In a perfect world this would be all that is necessary to estimate accurately the robot's pose at any time in the future. Unfortunately errors in the modeling (incorrect estimations of wheel size, vehicle size), uncertainty about the control inputs, realities of the motor controller (errors between commanded wheel rotation and true rotation), errors in the physical modeling of the robot (wheel compaction, ground

compaction, wheel slippage, nonzero tire width), etc., introduce an error between the dead reckoning estimate of the vehicle motion and its true motion. The problem of correcting for this error is the problem of *pose maintenance* for the vehicle, and requires the integration of the dead reckoning estimate with estimates obtained from other sensor systems.

Other chapters in this handbook (e.g., Chaps. 21–24) examine sensors that rely on external events, visual and otherwise, that can provide information as to the robot's pose or changes in its pose. Here we consider sensors that transduce physical properties of matter under the influence of external forces and properties of matter and the use of a global position system (GPS).

20.2 Gyroscopic Systems

The goal of gyroscopic systems is to measure changes in vehicle orientation by taking advantage of physical laws that produce predictable effects under rotation. A rotating frame is not an inertial frame, and thus many physical systems will appear to behave in an apparently *non-Newtonian* manner. By measuring these deviations from what would be expected in a Newtonian frame the underlying self-rotation can be extracted.

20.2.1 Mechanical Systems

Mechanical gyroscopes and gyrocompasses have a long history in navigation, *Bohnenberger* is generally credited with the first recorded construction of a gyroscope [20.4], and in 1851 Léon Foucault recognized the gyroscope as an inertial frame. The gyrocompass was patented in 1885 by Martinus Gerardus van den Bos. In 1903 Herman Anschütz-Kaempfe constructed a working gyrocompass and obtained a patent on the design. In 1908 Elmer Sperry patented a gyrocompass in the US and then attempted to sell this device to the German Navy. A patent battle followed, and Albert Einstein testified in the case. (See [20.5–8] for more details on the history of the gyrocompass and its inventors.)

Gyroscopes and gyrocompasses rely on the principle of the conservation of angular momentum [20.9]. Angular momentum is the tendency of a rotating object to keep rotating at the same angular speed about the same axis of rotation in the absence of an external torque. The angular momentum L of an object with moment of

inertia I rotating at angular speed ω is given by

$$L = I \times \omega.$$

Consider a rapidly spinning wheel mounted on a shaft so that it is free to change its axis of rotation (Fig. 20.2a). Assuming no friction due to air resistance or the bearings, the rotor axis will remain constant regardless of the motion of the external cage. This constancy of orientation can be exploited to maintain a bearing independently of the motion of the vehicle, although it is not usually desirable to use the principle of conservation of angular momentum via a gyroscope directly. To see this, suppose that a gyroscope is set on the equator, with its spinning axis aligned along the equator (Fig. 20.2b). As the Earth spins, the gyroscope will maintain a constant axis of orientation and thus to an Earth-fixed observer will appear to rotate, returning to its original orientation every 24 h. Similarly, if the gyroscope were to be positioned on the equator such that its spinning axis was parallel to the axis of rotation of the earth, the gyroscope's axis of rotation would remain stationary and would appear to remain stationary to an Earth-fixed observer as the planet rotates.

Although this global motion limits the mechanical gyroscope's ability to sense absolute orientation directly, gyroscopes can be used to measure local changes in orientation, and thus are well suited to vehicular robotic applications. Rate gyros (RGs) measure a vehicle's rotation rate (its *angular rate* of rotation). This is the fundamental measurement that is the basis of all gyroscopic systems. Rate-integrating

gyros (RIGs) use embedded processing to internally integrate the angular rotation rate to produce an estimate of the absolute angular displacement of the vehicle.

In order to exploit a gyroscope for navigation with respect to an Earth-stable frame, it is desirable for the rotational axis of the shaft to remain fixed within the Earth frame, rather than remaining fixed with respect to an external frame. A gyrocompass obtains this by relying on precession. When a torque is applied to change the axis of rotation of a spinning object, conservation of angular momentum will cause a change in the spin direction that is perpendicular to the angular momentum and the direction in which the torque is applied. This is the effect that causes gyroscopes suspended at one end to *spin* around the end from which they have been suspended. Consider the *pendulus gyro* sketched in Fig. 20.3a. This is a standard gyroscope with a weight suspended below the rotational axis. As before, imagine this pendulus gyrocompass set spinning on the equator with the axis of rotation aligned with the axis of rotation of the planet, and with the weight hanging directly down. As the planet spins, the gyroscope's axis of rotation remains stationary and would appear to remain stationary as the planet rotates. Now suppose that instead of being aligned with the spin axis of the planet the spin axis is aligned with the equator. As the planet spins, the spin axis is drawn out of the plane as it remains aligned with the original spin axis. As it becomes drawn out of the plane, the mass hanging below the gyrocompass is raised up and generates a torque down due to gravity. The direction perpendicular to the spin axis and the torque rotates the spin axis away from the equator known and towards the true pole. This process is sketched in Fig. 20.3b.

Unfortunately the pendulus gyro is not an ideal device for navigation. Although its rotation axis will align with the rotation axis of the planet, it does not converge to this value directly but oscillates about it. The solution to this damping problem is to use oil reservoirs, rather than a solid weight, as the counterbalance, and to restrict the motion of the oil in the reservoir [20.10].

The gyrocompass finds true north by controlling the precession of a gyroscope. In practice, the performance of a mechanical gyrocompass is impacted by external forces acting on the compass which also contribute to the precession of the gyroscope. This includes forces generated by motion of the device containing the gyrocompass, as well as any external forces acting on the vehicle. Another issue for mechanical gyrocompasses is that in latitudes away from the equator the stable position of the gyrocompass is not horizontal, and accurate estimates of true north at such latitudes requires corrections to be applied to raw gyrocompass values. Finally, mechanical gyrocompasses require an external force to be applied to maintain the spin of the gyroscope. This process introduces unwanted forces into the system which can further corrupt the measurement process.

Given the complexity, cost, size, the delicate nature of gyrocompasses, and the availability of less expensive and more reliable technologies, mechanical gyrocompasses have given way to optical- and microelectromechanical systems (MEMS)-based systems.

20.2.2 Optical Systems

Optical gyroscopes rely on the *Sagnac effect* rather than rotational inertia in order to measure (relative) heading. The mechanism is based on the behavior of an optical standing wave in a rotating frame. Historically this was first produced using lasers and an arrangement of mirrors, but it is now typically obtained using fibre-optic technology. The Sagnac effect is named after its discoverer Georges Sagnac [20.11, 12]. The underlying concept can be traced back even earlier to the work of Harress [20.13], and perhaps finds its most famous application in terms of the measurement of the rotation of the Earth [20.14].

Ignore relativistic effects and consider the circular light path shown in Fig. 20.4a. If two light pulses are sent in opposite directions around a stationary path of perimeter $D = 2\pi R$ they will travel the same distance at the same speed. They will arrive at the starting point simultaneously, taking time $t = D/c$ (where c is the speed of light in the medium). Now let us suppose that instead of being stationary, this circular light path rotates clockwise

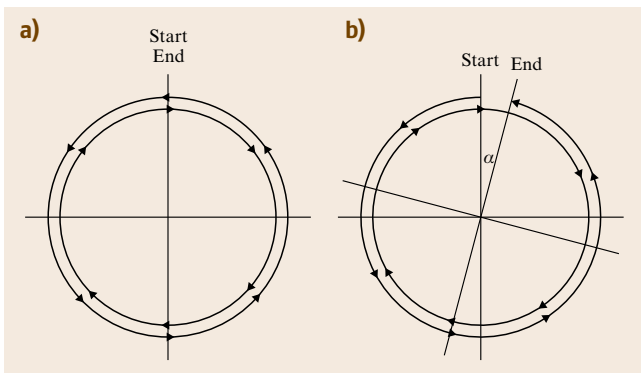


Fig. 20.4a,b Circular light path. (a) Stationary path. (b) Moving path

about its center at rotational speed ω (Fig. 20.4b). The light traveling clockwise around the path must go *farther* to reach the starting point, while light traveling counterclockwise around the path goes a *shorter* distance. The clockwise path has distance $D_c = 2\pi R + \omega R t_c$, where t_c is the time taken in the clockwise direction, while the counterclockwise path has distance $D_a = 2\pi R - \omega R t_a$, where t_a is the time taken in the counterclockwise direction. But $D_c = c t_c$ and $D_a = c t_a$, so $t_c = 2\pi R / (c - \omega R)$ and $t_a = 2\pi R / (c + \omega R)$. The time difference $\Delta t = t_c - t_a$ is given by

$$\Delta t = 2\pi R \left(\frac{1}{c - \omega R} - \frac{1}{c + \omega R} \right).$$

By measuring Δt , the rotational speed can be computed. Note that, although the above derivation assumes classical mechanics and ignores relativistic effects, the derivation also applies when relativistic speeds are taken into account [20.15]. See [20.16] for an in-depth review of the Sagnac effect and ring lasers.

In optical gyroscopes lasers are typically used as the light source. Optical gyroscopes either employ straight-line light paths with mirror surfaces or prisms at the edges to direct the light beam (a ring laser gyroscope – **RLG**), or a polarization maintaining glass-fiber loop (fiber optic gyro – **FOG**). The glass fiber may actually loop multiple times, thus extending the effective length of the light path. The time delay between the clockwise and counterclockwise directions is detected by examining the phase interference between the clockwise and counterclockwise light signals. Multiple optical gyroscopes with nonparallel axes can be ganged together in order to measure three-dimensional (3-D) rotations.

Various techniques can be used to measure the time difference between the two paths, including examining the Doppler (frequency) shift of the laser light due to the motion of the gyro and an examination of the beat frequency of the interference pattern between the clockwise and counterclockwise paths [20.17]. Ring interferometers typically consist of many windings of fiber-optic lines that conduct light of a fixed frequency in opposite directions around the loop and measure the phase difference. A ring laser consists of a laser cavity in the shape of a ring. Light circulates in both directions around this cavity, producing two standing waves with the same number of nodes in both directions. Since the optical path lengths are different in the two directions, the resonant frequencies differ. The difference between these two frequencies is measured. An unfortunate side-effect of the ring-laser approach is that the two signals will

lock in to each other for small rotations and it is typically necessary to physically rotate the device in a controlled manner in order to ensure that this lock-in effect can be avoided.

20.2.3 MEMS

Almost all micro-electromechanical systems (**MEMS**) gyroscopes are based on vibrating mechanical elements to sense rotation. Vibratory gyroscopes rely on the transfer of energy between vibratory modes based on Coriolis acceleration. Coriolis acceleration is the apparent acceleration that arises in a rotating frame of references. Suppose that an object moves along a straight line in a rotating frame of reference. To an outside observer in an inertial frame the object's path is curved – thus there must be some force acting on the object to maintain the straight line motion as viewed by the rotating observer. An object moving in a straight line with local velocity \mathbf{v} in a frame rotating at rate $\boldsymbol{\Omega}$ relative to an inertial frame will experience a Coriolis acceleration given by

$$\mathbf{a} = 2\mathbf{v} \times \boldsymbol{\Omega}.$$

Transducing acceleration in a **MEMS** gyroscope amounts to inducing some local linear velocity and measuring the resultant Coriolis forces.

Early **MEMS** gyroscopes utilized vibrating quartz crystals to generate the necessary linear motion. More recent designs have replaced the vibrating quartz crystals with silicon-based vibrators. Various **MEMS** structures have been developed including those described below.

Tuning-Fork Gyroscopes

Tuning-fork gyroscopes use a tuning-fork-like structure (Fig. 20.5) as the underlying mechanism. As the tuning forks vibrate within a rotating frame, Coriolis forces cause the tines of the fork to vibrate out of the plane of the fork, which is measured. This is the effect used by the InertiaCube sensor [20.18].

Vibrating Wheel Gyroscopes

Vibrating wheel gyroscopes use a wheel that oscillates about its rotational axis. External rotation of the frame causes the wheel to tilt, which can be sensed.

Wine-Glass Resonator Gyroscopes

Wine-glass resonator gyroscopes use the effect of Coriolis forces on the position of nodal points on a resonating structure to estimate the external rotation.

As **MEMS** gyroscopes have no rotating parts, have low-power consumption requirements, and are very

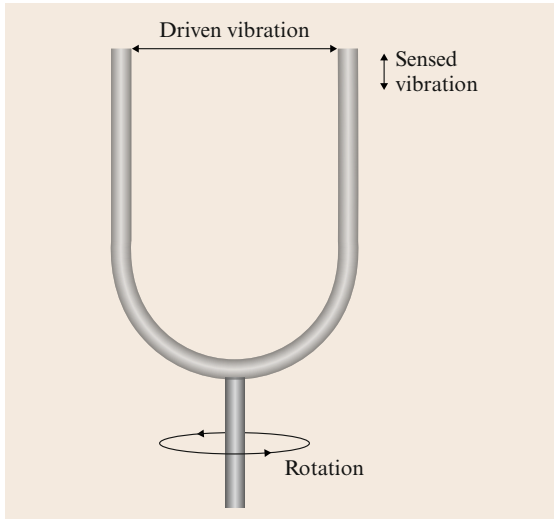


Fig. 20.5 MEMS gyroscope: principle of operation

small, MEMS gyros are quickly replacing mechanical and optical gyroscope sensors in robotic applications.

20.2.4 Performance

Inertial measurement units can be evaluated with respect to various factors that determine performance, several of which are enumerated below.

1. Bias repeatability. This is the maximum deviation of the gyroscope under fixed inertial operation conditions with constant temperature, i. e., the drift of the reading under ideal conditions. This is measured over different time scales and leads to *short-term* and *long-term* bias repeatability.
2. Angle random walk. This measures the noise in the angular rate data coming from the gyro.

20.3 Accelerometers

Just as gyroscopes can be used to measure changes in orientation of a robot, other inertial sensors – known as *accelerometers* – can be used to measure external forces acting on the vehicle. One important factor concerning accelerometers is that they are sensitive to all external forces acting upon them – including gravity. Accelerometers use one of a number of different mechanisms

3. Scale factor ratio. This parameter is not specific to inertial measurement units (IMUs) or gyros and is a general measurement of signal amplitude. It measures the ratio of the output analogue voltage to the sensor parameter of interest. For a gyroscope this is typically measured in $\text{mV}/(\text{deg}/\text{sec})$ whereas for an accelerometer it is typically measured in $\text{mV}/(\text{m}/\text{s}^2)$.

20.2.5 Summary

With the exception of gyrocompasses, gyroscopes measure relative rotational motion around a single axis. They accomplish this measurement by exploiting physical properties of rotating frames of reference. Earlier technologies based on mechanical gyroscopes have given way to optical- and MEMS-based devices but the underlying principle remains unchanged: that rotating frames of reference show specific physical properties that can be measured to estimate the relative rotation.

A problem common to all gyroscopes is that of drift. Each of the relative motion measurements is corrupted by an error process, and these errors accumulate over time. This, coupled with specific measurement errors associated with the individual gyroscope technologies, means that unless the error is corrected through reference to some alternate (external) measurement, the drift will eventually exceed the required accuracy of the measurement.

As individual gyros only measure rotation about a single axis, it is common to gang multiple gyros together with orthogonal axes of sensitivity in order to measure 3-D rotations. These collections of gyros are often integrated with other sensors (compasses, accelerometers, etc.) in order to construct *inertial measurement units* (or IMUs). This is considered in Sect. 20.4.

to transduce external forces into a computer-readable signal.

Mechanical Accelerometer

A mechanical accelerometer (Fig. 20.6a) is essentially a spring–mass–damper system with some mechanism for external monitoring. When some force is applied

(e.g., gravity), the force acts on the mass and displaces the spring. Assuming an ideal spring with a force proportional to its displacement, the external forces balance the internal ones

$$\begin{aligned} F_{\text{applied}} &= F_{\text{inertial}} + F_{\text{damping}} + F_{\text{spring}} \\ &= m\ddot{x} + c\dot{x} + kx, \end{aligned}$$

where c is the damping coefficient. This equation can be solved to show that, depending on the size of the damping coefficient relative to the expected external force and the mass, the system can be made to reach a stable final value in a reasonably short period of time whenever a static force is presented. This need to pre-estimate the expected force and the resulting (potentially long) time for the system to converge on a final measurement coupled with nonideal performance of the spring limits the applicability of mechanical accelerometers. Another issue with mechanical accelerometers is that they are particularly sensitive to vibration.

Piezoelectric Accelerometer

Rather than relying on a direct mechanical measurement of external forces, piezoelectric accelerometers are based on a property exhibited by certain crystals, across which a voltage is generated when they are stressed. A small mass can be positioned so that it is only sup-

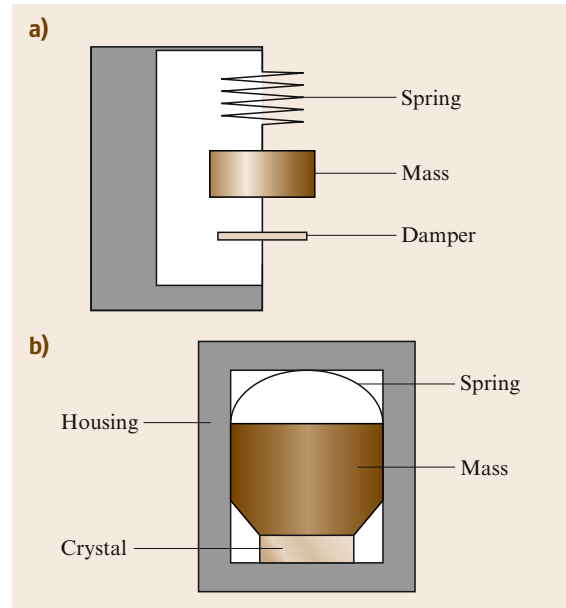


Fig. 20.6a,b Accelerometers. (a) Mechanical accelerometer. (b) Piezoelectric accelerometer

ported by the crystal, and as forces cause the mass to act upon the crystal this induces a voltage that can be measured (Fig. 20.6(b)).

20.4 IMU Packages

An inertial measurement unit (IMU) is a device that utilizes measurement systems such as gyroscopes and accelerometers to estimate the relative position, velocity, and acceleration of a vehicle in motion. The resulting navigation system is known as an inertial navigation system or INS. First demonstrated in 1949 by C. S. Draper, IMUs have become a common navigational component of aircraft and ships. Historically an IMU is self-contained and provides this estimate without reference to external references, however the definition has become less precise in recent years and now it is common to apply the term IMU also to systems that do include such external references.

IMUs come in two basic flavors, gimbaledd systems and strap-down systems. As their name suggests, gimbaledd IMUs are mounted within complex gimbal structures in order to provide a stable platform from which measurements can be made. Gyroscopes are used to ensure that the gimbal remains aligned with the initial reference frame at power up. The orienta-

tion of the gimbaledd platform relative to the vehicle is used to map measurements taken within the IMU to the reference frame of the vehicle. Strap-down IMUs, on the other hand, have the IMU rigidly connected to the vehicle (strapped down), so no such transformation is required. In either case estimating the motion relative to the initial frame requires integrating information from the sensors within the IMU (accelerometers, gyroscopes, etc.) in real time. This was a significant computational effort in the early days of IMUs, and thus historically (prior to the 1970s) the gimbaledd IMU was more common. Given the low cost of such computation today, and the costs associated with manufacturing and operating gimbaledd IMUs, strap-down IMUs are much more common today [20.19].

A true IMU maintains a 6-degree-of-freedom (DOF) estimate of the pose of the vehicle: position (x , y , z) and orientation (roll, pitch, yaw). IMU-like systems that (for example) only maintain ongoing estimates of orientation

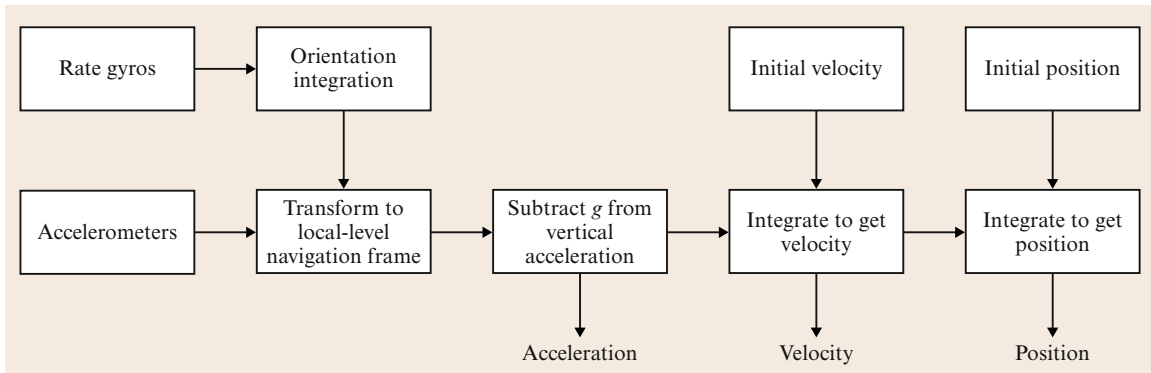


Fig. 20.7 IMU block diagram

are known as *attitude and heading reference systems*, and operate in a similar fashion as IMUs but maintain a less full representation of the vehicle's state. In addition to maintaining a 6-DOF pose of the vehicle, commercial IMUs also typically maintain estimates of velocity and acceleration.

The basic computational task of an IMU is shown in Fig. 20.7. This IMU uses three orthogonal accelerometers and three orthogonal gyroscopes. The gyroscope data ω is integrated to maintain an ongoing estimate of vehicle orientation θ . At the same time, three accelerometers are used to estimate the instantaneous vehicle acceleration a . This data is then transformed via the current estimate of the vehicle orientation relative to gravity, so that the gravity vector can be estimated and extracted from the measurement. The resulting accel-

ation is then integrated to obtain vehicle velocity v and then integrated again to obtain the position r .

IMUs are extremely sensitive to measurement errors in the underlying gyroscopes and accelerometers. Drift in the gyroscopes leads to misestimates of the vehicle orientation relative to gravity, resulting in incorrect cancellation of the gravity vector. As the accelerometer data is integrated twice, any residual gravity vector will result in a quadratic error in position [20.18]. As it is never possible to eliminate the gravity vector completely, and this and any other error is integrated over time, drift is a fundamental issue for any IMU. Given a sufficiently long period of operation all IMUs eventually drift and reference to some external measurement is required to correct this. For many field robots GPS has become an effective source for these external corrections.

20.5 GPS

The global positioning system (GPS) is the single most commonly used mechanism for location estimation. It provides a three-dimensional position estimate in absolute coordinates as well as current time and date and is available anywhere on the Earth's surface. Standard GPS provides a position estimate in the horizontal plane to within about 20 m. It was originally developed for military applications but has become widely adopted in civilian applications, including automobile navigation systems, recreational orienteering, and inventory tracking for transportation companies.

20.5.1 Overview

The system is based on received radio signals transmitted by an ensemble of satellites orbiting the Earth.

By comparing the time delays from the different satellite signals, a position fix can be computed. The most widely accepted GPS system is based on the NAVSTAR satellite system deployed and maintained by the United States, specifically by the Air Force Space Command. As a US military service, the US government reserves the right to terminate or modify its availability at their discretion. A similar system named *Globalnaya Navigatsionnaya Sputnikovaya Sistema (GLONASS)* is operated by the Russian government, but at the time of writing it is not available for robotic applications. Another alternative system is being deployed by the European Union, named *Galileo*, which has the explicit expectation that it will not be under military control. It is expected to offer two different classes of service: an open service and an encrypted higher-quality commercial service. Other GPS systems

such as the Chinese *Beidou* or the Japanese *QZSS* are outside the scope of this chapter.

In general the term **GPS** always refers to the NAVSTAR system. Historically NAVSTAR provided two different services: the precise positioning system (PPS), reserved primarily for military users, and the standard position system (SPS) with lower accuracy. The difference between these was referred to as selective availability (SA). This difference in accuracy was artificially induced via pseudorandom noise in the SPS signal for strategic reasons, and was eliminated in 2001. Although this accuracy distinction could, in principle, be reinstated this seems unlikely due to the widespread commercial dependence on GPS that has arisen in the last decade. Even now, the GPS network includes supplementary data in encrypted precision P(Y) code which is not available to civilian users.

The GPS satellite network is based on a base constellation of 24 orbiting satellites along with up to six supplementary additional satellites that are also operational. These satellites are in almost-circular medium Earth orbit. As opposed to being geostationary the orbits are semisynchronous, meaning that their position relative to a ground observer is constantly changing, and that their orbital period is exactly half a sidereal day. The orbits are selected so that from almost any point of the Earth's surface there will always be four or more satellites directly visible – a criterion for obtaining a GPS position estimate. The satellites are organized into six orbital planes with four satellites in each. The system is designed such that averaged over the entire Earth's surface and over a 24 h interval, the satellites should allow 99.9% coverage at the worst-covered location in any 24 h interval, and the signal should be at least 83.92% available at the worst place on Earth on the worst day over a 30-day measurement interval. Note that this criterion for availability takes into account transmitted operational factors above and beyond simple coverage. Of course, this criterion ignores the reality of topographic features like mountains, as well as other objects such as buildings that can obstruct the line of sight.

Each satellite repeatedly broadcasts a data packet known as the coarse-acquisition (C/A) code, which is received by the GPS receiver on the L1 channel at 1575.42 MHz. The simple principle is that, if the receiver knows the absolute positions of the observed satellites, the receiver position can be directly determined. If the signal propagation time for the radio signals were known, the receiver position could be computed directly via trilateration (Fig. 20.8). This implies that an abso-

lute timing reference is present on the receiver, which would be prohibitively costly. Instead, only the satellites have highly accurate atomic clocks (accurate to approximately 1 s in 300 000 years). The receiver computes the difference in signal propagation times between the different satellites, and uses this to compute a range estimate referred to as a *pseudorange* (to explicitly indicate that it is corrupted by several sources of measurement noise). The specific geometric problem is referred to as *multilateration* or *hyperbolic positioning* and the solution is computed using a sophisticated Kalman filter within the GPS receiver. To avoid retaining an ephemeris (pose) table for the satellites and a very accurate clock in the receiver, each satellite broadcasts its own position and an accurate time signal as part of the data packet that it transmits.

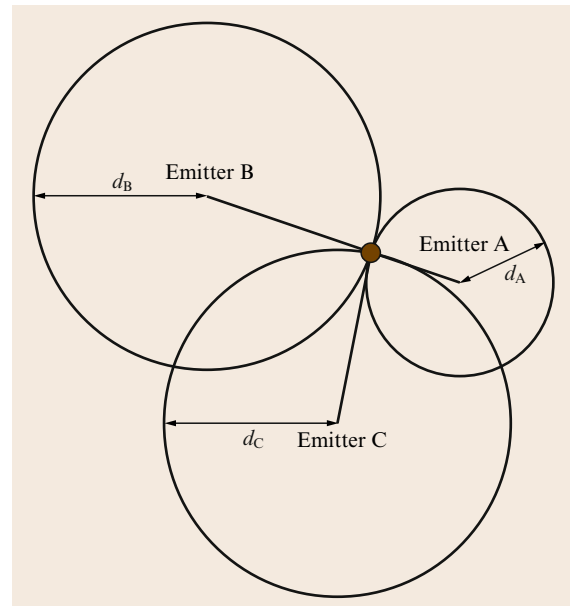


Fig. 20.8 GPS trilateration on the plane. Suppose that one receives signals from three transmitters (A, B, and C) with known locations. Knowledge of the signal delay from one emitter (say A) localizes the receiver to lie on a circle of known diameter (d_A) whose center is the emitter. The constraints from two emitters intersect at two points (maximum). A third emitter is required to disambiguate these two solutions. In three dimensions, the signal propagation constraint from a single emitter constrains the receiver to lie on a sphere. The intersection of the constraints from two emitters constrains the receiver to lie on a circle. The intersection of the constraints from three emitters constrains the receiver to one of two points

GPS satellites broadcast at several different frequencies known as L1 through L5; only L1 (1575.42 MHz) and L2 (1227.6 MHz) are used by civilian GPS receivers. The standard service offered by NAVSTAR and the performance criteria for it are determined by the L1 signal, which contains two unencrypted components: the acquisition message (coarse-acquisition message C/A) and a navigation data message. It is also possible to use the encrypted L2 signal as well, even without the secret decryption keys, to provide augmented error correction (by observing the relative effects of ionospheric distortion as a function of frequency). The restricted-access signal broadcast on both the L1 and L2 channels is the P-code (as well as a fairly recent M-code) which is known as the Y-code or P(Y) or P/Y code once it is encrypted. Both the C/A and P(Y) codes include the navigation message stream that specifies clock bias data, orbital information, ionospheric propagation corrections factors, ephemeris data, status information on all the satellites, universal time code, and other information. The satellite performance is coordinated by the master control station located at the Schriever Air Force Base near Colorado Springs, Colorado, USA and is connected to a global network of five additional monitoring stations (Cape Canaveral USA, Ascension Island, Kwajalein Atoll in the Marshall Islands, Diego Garcia Atoll, and Hawaii) which are used produce the measurements that are uplinked to generate the navigation message stream. Finally, it should be noted that an additional signal is now available on the L2 frequency band. This L2C signal on satellites designated *block IIR-M* promises to provide much improved receiver sensitivity so that position fixes can be obtained in environments, such as in forests, where they are currently not readily available.

20.5.2 Performance Factors

GPS performance depends on several factors: satellite transmission accuracy, environmental conditions, interactions with ground-based obstacles, and receiver properties.

In the context of robotics, factors that affect the performance of the satellites themselves and the atmospheric conditions are essentially uncontrollable. Nevertheless, it should be noted that these can be sources of error and that the GPS signal itself may not always be reliable. A *service failure* is defined as a set of circumstances where the positioning service exhibits atypical error behavior (i. e., incorrect signals). Such failures are classified as either minor and major failures. Minor fail-

ures are those that have limited impact on a receiver and lead to ranging errors of under 150 m. Major failures are those that lead either to larger errors or data processing overloads in the receiver. If a single satellite experiences an error that leads to a major failure, then within a 6 h period, approximately 63% of the Earth's surface will have the satellite in view at some point.

The controllable factors in using GPS for accurate localization are

1. it requires an unobstructed line of sight to the satellites,
2. it depends on atmospheric conditions, and
3. it depends on the ability to receive (weak) radiofrequency communications.

There is a potential for wildly incorrect estimates. Generally satellites that are directly overhead provide better signals than those near the horizon. In addition, since the basis of GPS position is differential signal analysis, it is best if the satellites used in the GPS computation are widely spaced in the sky.

GPS signals are in the microwave band and, as such, they can pass through plastic and glass, but are absorbed by water (wood, heavy foliage) and are reflected by many materials. As a consequence, GPS is unreliable in heavy forest, deep canyons, inside automobiles and boats, in heavy snowfall or between tall buildings. In some cases, partial obstruction of the sky may not prevent a position estimate from being computed. Assuming the minimum number of satellites operating at any time is 24, then on average across the Earth's surface eight satellites are in view so that even partial occlusion of the sky can often be tolerated. On the other hand, partial occlusion can lead to reduced accuracy since the selection of available satellites used for computing position becomes limited and optimal accuracy is obtained by using as many satellites as possible (weighting them appropriately in the internal Kalman filter).

Secondary factors that differentiate different GPS receivers are the rate at which the signals are collected, the receiver sensitivity, the number of satellites used in the final computation, the number of factors taken into account in the estimator, and the exploitation of supplementary positioning schemes such as the wide-area augmentation system (WAAS) (see later). A major factor in determining the rate at which estimates can be produced is the number of independent receiver elements in the GPS system. Sequential single-channel receivers are simpler and thus more economical (and potentially smaller), but they must lock sequentially onto each satellite being used. Parallel multichannel receivers can lock

onto to more than one satellite at once, and are generally faster and more costly; some degree of parallelism is the norm in good-quality consumer devices.

GPS computations are based on an estimation of the dilution of precision (DOP) and specifically for the dilution of precision of the positional parts of the system, i. e., positional dilution of precision (PDOP). These correspond to partial derivatives of the error with respect to position and allow the most accurate ensemble of visible satellites to be determined at any time. The standard implementation for GPS systems specifies that PDOP values be recomputed every 5 min.

The minimum performance parameters for GPS receivers are based upon transforming instantaneous range residuals to a user position estimate using a linearized position solution from a stationary, surveyed location. Most GPS receivers use additional techniques such as range residual smoothing, velocity aiding, Kalman filtering, or multiple satellite (*all-in-view satellite*) solutions. That said, formal performance for the system is measured with respect to the minimum. The GPS position estimation algorithm is summarized as follows

1. Select the best four satellites based upon the minimum error measured in terms of PDOP.
2. Update every five minutes, or whenever a satellite being used in the solution sets.
3. Measure the pseudorange to each satellite. Each of the four measurements must have a reception time tag within 0.5 s of the solution time. The reception time tag is based upon measurement system time, and the transmission time tag is based upon satellite time.
4. Determine the ephemeris for each of the satellites being used, and compute the Earth-centered, Earth-fixed (ECEF) coordinates for each. Correct for the Earth's rotation and thus compute an estimated pseudorange measurement that should be obtained for each satellite.
5. Compute the *range residuals* as the differences between the actual and observed measurements.
6. Estimate the matrix G that determines the overall system solution, known as the *position solution geometry matrix*. The matrix can be described in terms of a collection of row vectors, one for each of the satellites being used, each row being made up of the x , y , z and time coordinate direction cosines for the vector between the user and the satellite (with respect to a fixed reference frame for the planet called the World Geodetic System, WGS84).
7. Compute the user's position.

The standard implementation of GPS is based upon a position fix rate of once per second, although faster and slower rates are possible. Under typical operating conditions and without specialized enhancements GPS accuracy is roughly 20–25 m in the horizontal plane and 43 m in the vertical direction. The restricted PPS signal provides an accuracy of at least 22 m (typical values are 7–10 m) in the horizontal plane and 27.7 m in the vertical direction as well as coordinated universal time (UTC) time accuracy within 200 ns based on a reference signal from the US Naval Observatory.

GPS signals can be affected by multipath issues, where the radio signals reflect off surrounding terrain – buildings, canyon walls, hard ground, etc. This delay in reaching the receiver causes inaccuracy. A variety of receiver techniques, most notably narrow correlator spacing [20.20], have been developed to mitigate multipath errors. For long delay multipaths, the receiver itself can recognize the wayward signal and discard it. To address shorter delay multipaths due to the signal reflecting off the ground, specialized antennas may be used. This form of multipath is harder to filter out as it is only slightly delayed compared to the direct signal, causing effects that are almost indistinguishable from routine fluctuations in the atmospheric delay.

20.5.3 Enhanced GPS

Wide-Area Augmentation System (WAAS)

The wide-area augmentation system (WAAS) is a supplementary signal that can be received by GPS receivers to improve their accuracy. WAAS increases the accuracy of horizontal position estimates from 10–12 m with GPS alone, to 1–2 m. The WAAS signal contains corrections for the GPS signal that reduce the effects of errors due to timing errors, satellite position corrections, and local perturbations due to variations in the ionosphere. These correction terms are estimated by ground-based stations at fixed and accurately known positions and uplinked to satellites which broadcast them to suitably enabled GPS receivers. The WAAS signal is only computed and available for North America, but similar correction signals are becoming available elsewhere as part of the standardization of satellite-based augmentation systems (SBAS). This includes Europe, where it is called the Euro Geostationary Navigation Overlay Service (EGNOS), and Japan and parts of Asia, where it is called the Multifunctional Satellite Augmentation System (MSAS). Further enhancements to GPS and WAAS, in the form of the Global Navigation Satellite System Landing System (GLS), are slated for completion in 2013.

Differential GPS

Differential GPS is a technique for correcting GPS signals by using a nearby GPS receiver located at a known accurately surveyed position. In fact, several variations on this basic idea exist and are also known under the general rubric of ground-based augmentation systems (GBAS). DGPS uses the same principles as WAAS but on a local scale without resorting to the use of satellite uplinks. The receiver at the known position computes the error in the GPS signal and transmits it to the nearby receiver at the unknown location. Since the error varies as a function of position on the earth, the effectiveness of the correction degrades with distance, typically with a maximum effective range of a couple of hundred miles. The method was especially desirable before the suspension of selective availability and the development of WAAS (which can be viewed as a form of DGPS). In the USA and Canada, a network of ground-based DGPS transmitters are in place, sending signals using radio frequencies between 285 kHz and 325 kHz. Commercial DGPS solutions akin to WAAS also exist.

Receiver Autonomous Integrity Monitor (RAIM)

Receiver autonomous integrity monitoring (RAIM) is a technique by which multiple pseudorange measurements (i. e., pose estimates) are obtained using different combinations of satellites. If inconsistent measurements are obtained, it indicates that a failure of some sort has taken place in the system. A position fix using at least five satellites is needed to detect such an error while at least six satellite are needed to exclude the data from a single bad satellite and still obtain a reliable estimate.

20.5.4 GPS Receivers and Communications

GPS receivers are classified according to their performance and cost. The best receivers are referred to as *geodetic grade* with economical models referred to as *resource grade* or *recreational*. In general the costs of these

different models vary by several orders of magnitude, but the gap in performance is gradually narrowing.

Receivers come in two types: code phase and carrier phase. Code phase receivers use the satellite navigation message part of the data stream to provide the ephemeris data and produce real-time output. There is a delay for them to lock into the satellites, but then they produce output continuously without an initial position estimate. The C/A signal is a 1023 pseudorandom *noise* (PRN) bit string with a known key. The actual pseudorange data is determined by finding the phase offset of this bit string.

Carrier phase receivers, on the other hand, use the phase of the raw GPS signal rather than the embedded (digital) C/A signal. The L1 and L2 signals have wavelengths of 19 and 24 cm, respectively, and good-quality phase measurements allow horizontal positioning accuracies on the order of millimeters. These measurements, however, only provide relative position information within a neighborhood of some tens of kilometers.

Serial Protocols

Consumer GPS devices almost universally support some variant of the National Marine Electronics Association (NMEA) protocol, a serial protocol often transmitted using RS-232 wiring. Several variants of the protocol exist, but NMEA 0183 is the most commonly supported while NMEA 2000 supports higher data rates. While the protocol is proprietary and the official specification can only be purchased from the National Marine Electronics Association, there are several open-source descriptions of the protocol that have been reverse engineered.

The protocol supports an ASCII mode of communications based on a *talker* (the GPS receiver) and one or more *listeners* (computers) which receive simple protocol strings called sentences. Anecdotal evidence suggests that there are ambiguities in the protocol that can lead to difficulties in assuring compatibility between devices (this suggestion is necessarily made without inspecting the proprietary documentation, which might preclude its description here).

20.6 GPS-IMU Integration

Although GPS offers the promise of high-resolution positioning information on or about the surface of the Earth, it does not solve all of the problems associated with robot pose estimation. First, it does not directly obtain information about vehicle orientation. To determine the orientation of the vehicle yaw, and for many vehicles

pitch and roll, must be estimated by either differentiating the GPS signal or by integration with other sensors such as compasses, gyrocompasses, and IMUs. Second, GPS receivers are generally unable to provide continuous independent estimates of position. Estimates are only available at distinct time instances with (for inex-

pensive receivers at least) considerable delays between measurements. A continuous estimate of pose requires estimation of pose between GPS readings. Finally, it is not always possible to obtain a GPS fix. Local geography (e.g., mountains, buildings, trees) or an overhead cover that is opaque to radio signals (e.g., indoors, underwater) can block the signal entirely. Integration of a GPS receiver with another sensor technology (often an IMU) can be used to deal with these issues, at least for short periods of time.

The process of integrating GPS and IMU data is typically expressed as a Kalman filter estimation process (Sect. 25.2.3). Essentially the IMU data is used to bridge between solid GPS measurements and is combined in a least-squares optimal sense with the GPS data when both are available. Given the complementary nature and true independence of the two sensors, a wide range of commercial packages have been developed to integrate GPS and IMU data (see [20.21] for an example).

20.7 Further Reading

Odometry

Many general robotics books, including [20.2] and [20.22], provide considerable information on vehicle odometry and derive odometry equations for standard vehicle designs.

Gyroscopic Systems

Everett's book [20.23] provides a review of various sensor technologies including gyroscopic systems and accelerometers. Interesting historical documentation on the gyrocompass and its inventor can be found in Hughes's book [20.5].

Accelerometers

Everett's book [20.23] provides a review of various sensor technologies, including gyroscopic systems and accelerometers.

GPS

Considerable details on the theory and implementation of GPS systems can be found in Leick's book [20.24]. See also [20.25]. Details of various approaches to GPS-INS integration can be found in [20.26] and [20.27].

20.8 Currently Available Hardware

Although the specific models listed below are likely to have a short shelf life, the list of contacts may be a good starting point for the identification of specific inertial sensing devices.

Gyroscopic Systems

- KVN DSP-3000 Tactical Grade Fiber Optic Gyro (FOG).
KVH Industries Inc., 50 Enterprise Center,
Meddletown, RI 02842-5279, USA
- Fiber Optic Gyroscope HOFG-1(A).
Corporate Headquarters Hitachi Cable Ltd.,
4-14-1 Sotokanda, Chiyoda-ku, Tokyo 101-8971,
Japan
- Rate Gyroscope CRS03,
Silicon Sensing Systems Japan Ltd.,
1-10 Fusocho (Sumitomo Precision Complex),
Amagasaki, Hyogo 660-0891, Japan

Accelerometers

- Accelerometer FA 101,
A-KAST Measurements and Control Ltd., 1054-2
Centre St. Suite #299,
Thornhill, ON, L4J 8E5, Canada
- ENDEVCO MODEL 22,
Brüel and Kjær,
DK-2850 Naerum, Denmark

IMU Packages

- μ IMU,
MEMSense, 2693D Commerce Rd.,
Rapid City, SD 57702, USA
- IMU400 MEMS Inertial Measurement Unit,
Crossbow Technology Inc.,
4145 N. First St., San Jose, CA 95134, USA
- IntertiaCube3, (3DOF IMU),
Intersense,

36 Crosby Dr, #15,
Bedford, MA 01730, USA

GPS Components

- Garmin **GPS** 18
Garmin International Inc.,
1200 East 151st St.,
Olathe, KS 66062-3426, USA

- Magellan Meridian Color
Thales Navigation
471 El Camino Real,
Santa Clara, CA 95050-4300, USA
- TomTom Bluetooth **GPS** Receiver
Rembrandtplein 35,
1017 **CT** Amsterdam,
The Netherlands

References

- 20.1 Vitruvius: *Vitruvius: The Ten Books on Architecture* (Harvard Univ. Press, London 1914), English translation by M. Morgan
- 20.2 G. Dudek, M. Jenkin: *Computational Principles of Mobile Robotics* (Cambridge Univ. Press, Cambridge 2000)
- 20.3 E. Maloney: *Dutton's Navigation and Piloting*, 14th edn. (US Naval Inst. Press, Annapolis 1985)
- 20.4 J.G.F. Bohnenberger: Beschreibung einer Maschine zur Erläuterung der Gesetze der Umdrehung der Erde um ihre Achse, und der Veränderung der Lage der Letzteren, *Tübinger Bl. Naturwiss. Arzneikunde* **3**, 72–83 (1817), (in German)
- 20.5 T.P. Hughes: *Elmer Sperry: Inventor and Engineer* (The Johns Hopkins Univ. Press, Baltimore 1993), Reprint Edition
- 20.6 H.W. Sorg: From Serson to Draper – two centuries of gyroscopic development, *Navigation* **23**, 313–324 (1976)
- 20.7 W. Davenport: *Gyro! The Life and Times of Lawrence Sperry* (Scribner, New York 1978)
- 20.8 J.F. Wagner: From Bohnenberger's machine to integrated navigation systems, 200 years of inertial navigation, *Photogramm. Week* **05**, 123–134 (2005)
- 20.9 R.P. Feynman, R.B. Leighton, M. Sands: *The Feynman Lectures on Physics* (Addison-Wesley, Reading 1963)
- 20.10 Navpers 16160: The Fleet Type Submarine. Produced for ComSubLant by Standards and Curriculum Division Training, Bureau of Naval Personnel, United States Navy (1946)
- 20.11 G. Sagnac: L'ether lumineux demontre par l'effect du vent relatif d'ether dans un interferometre en rotation uniforme, *C. R. Acad. Sci. Paris* **157**, 708–710 (1913), (in French)
- 20.12 G. Sagnac: Sur la preuve de la realitet de l'ether lumineux par l'experience de l'interferographie tournant, *C. R. Acad. Sci. Paris* **157**, 1410–1413 (1913), (in French)
- 20.13 F. Harress: Die Geschwindigkeit des Lichtes in bewegten Körpern, Dissertation (Friedrich-Schiller-Universität, Jena 1912) (in German)
- 20.14 A.A. Michelson, H.G. Gale: The effect of the earth's rotation on the velocity of light, *J. Astrophys.* **61**, 140–145 (1925)
- 20.15 S. Ekeziel, H.J. Arditty: *Fiber-Optic Rotation Sensors*, Springer Ser. Opt. Sci., Vol. 32 (Springer, Berlin, Heidelberg 1982)
- 20.16 G.E. Stedman: Ring-laser tests of fundamental physics and geophysics, *Rep. Prog. Phys.* **60**, 615–688 (1997)
- 20.17 D. Mackenzie: From the luminiferous ether to the Boeing 757: A history of the laser gyroscope, *Technol. Cult.* **34**, 475–515 (1993)
- 20.18 E. Foxlin, M. Harrington, Y. Altshuler: Miniature 6-DOF inertial system for tracking HMDs, *Proc. SPIE* **3362**, 214–228 (1998)
- 20.19 M. Mostafa: History of inertial navigation systems in survey applications, *J. Am. Soc. Photogramm. Remote Sens.* **67**(12), 1225–1227 (2001)
- 20.20 A.J. Van Dierendonck, P. Fenton, T. Ford: Theory and performance of narrow correlator spacing in a GPS receiver, *Navigation J. Inst. Navigation* **39**, 265–283 (1992)
- 20.21 J. Rios, E. White: Low cost solid state GPS/INS package, *Proc. Inst. Navigation Conf.* (2000)
- 20.22 R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2004)
- 20.23 H.R. Everett: *Sensors for Mobile Robots: Theory and Application* (Peters, Wellesley 1995)
- 20.24 A. Leick: *GPS Satellite Surveying* (Wiley, New York 2004)
- 20.25 P. Misra, P. Enge: *Global Positioning System: Signals, Measurements, and Performance*, 2nd edn. (Ganga-Jamuna, Lincoln 2006)
- 20.26 J. Farrell, M. Barth: *The Global Positioning System and Inertial Navigation* (McGraw-Hill, New York 1989)
- 20.27 M.S. Grewal, L.R. Weill, A.P. Andrews: *Global Positioning Systems, Inertial Navigation, and Integration* (Wiley-Interscience, New York 2007)

Sonar Sensin

21. Sonar Sensing

Lindsay Kleeman, Roman Kuc

Sonar or ultrasonic sensing uses the propagation of acoustic energy at higher frequencies than normal hearing to extract information from the environment. This chapter presents the fundamentals and physics of sonar sensing for object localization, landmark measurement and classification in robotics applications. The source of sonar artifacts is explained and how they can be dealt with. Different ultrasonic transducer technologies are outlined with their main characteristics highlighted.

Sonar systems are described that range in sophistication from low-cost threshold-based ranging modules to multitransducer multipulse configurations with associated signal processing requirements capable of accurate range and bearing measurement, interference rejection, motion compensation, and target classification. Continuous-transmission frequency-modulated (CTFM) systems are introduced and their ability to improve target sensitivity in the presence of noise is discussed. Various sonar ring designs that provide rapid surrounding environmental coverage are described in conjunction with mapping results. Finally the chapter ends with a discussion of biomimetic sonar, which draws inspiration from animals such as bats and dolphins.

21.1	Sonar Principles	492
21.2	Sonar Beam Pattern	494
21.3	Speed of Sound	496
21.4	Waveforms	496
21.5	Transducer Technologies	497
21.5.1	Electrostatic	497
21.5.2	Piezoelectric	498
21.5.3	MEMS	498
21.6	Reflecting Object Models	499
21.7	Artifacts	500
21.8	TOF Ranging	501
21.9	Echo Waveform Coding	503
21.10	Echo Waveform Processing	506
21.10.1	Ranging and Wide-Bandwidth Pulses	506
21.10.2	Bearing Estimation	506
21.11	CTFM Sonar	508
21.11.1	CTFM Transmission Coding	508
21.11.2	CTFM TOF Estimation	508
21.11.3	CTFM Range Discrimination and Resolution	509
21.11.4	Comparison of CTFM and Pulse-Echo Sonar	509
21.11.5	Applications of CTFM	510
21.12	Multipulse Sonar	511
21.12.1	Interference Rejection	511
21.12.2	On-the-Fly Target Classification	511
21.13	Sonar Rings	512
21.13.1	Simple Ranging Module Rings	512
21.13.2	Advanced Rings	513
21.14	Motion Effects	513
21.14.1	Moving Observation of a Plane	513
21.14.2	Moving Observation of a Corner	514
21.14.3	Moving Observation of a Edge	514
21.14.4	The Effect of a Moving Observation on the Angle of Reception	515
21.14.5	Plane, Corner, and Edge Moving Observation Arrival Angles	515
21.15	Biomimetic Sonars	515
21.16	Conclusions	516
	References	517

21.1 Sonar Principles

Sonar is a popular sensor in robotics that employs acoustic pulses and their echoes to measure the range to an

object. Since the sound speed is usually known, the object range is proportional to the echo travel time. At ultrasonic frequencies the sonar energy is concentrated in a beam, providing directional information in addition to range. Its popularity is due to its low cost, light weight, low power consumption, and low computational effort, compared to other ranging sensors. In some applications, such as in underwater and low-visibility environments, sonar is often the only viable sensing modality.

Sonars in robotics have three different, but related, purposes:

1. **Obstacle avoidance:** the first detected echo is assumed to measure the range to the closest object. Robots use this information to plan paths around obstacles and to prevent collisions.
2. **Sonar mapping:** a collection of echoes acquired by performing a rotational scan or from a sonar array is used to construct a map of the environment. Similar to a radar display, a range dot is placed at the detected range along the probing pulse direction.
3. **Object recognition:** a sequence of echoes or sonar maps is processed to classify echo-producing structures composed of one or more physical objects. When successful, this information is useful for robot registration or landmark navigation.

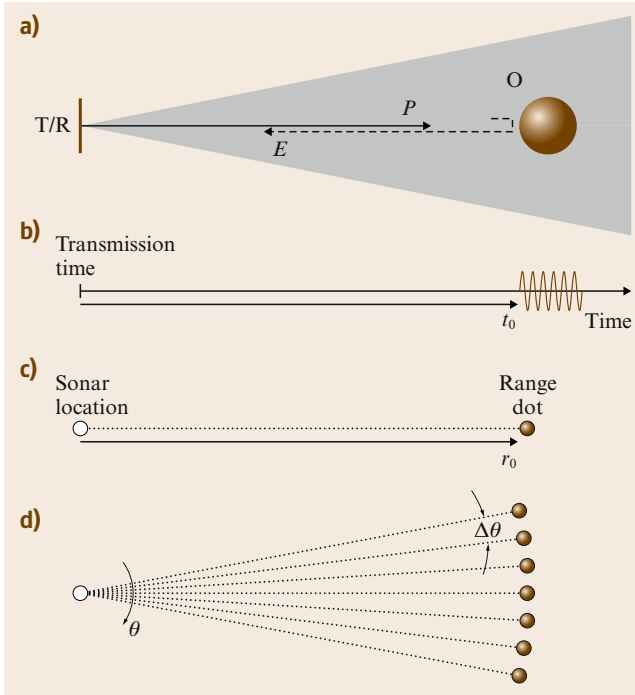


Fig. 21.1a–d Sonar ranging principles: (a) sonar configuration, (b) echo waveform, (c) range dot placement, (d) sonar map

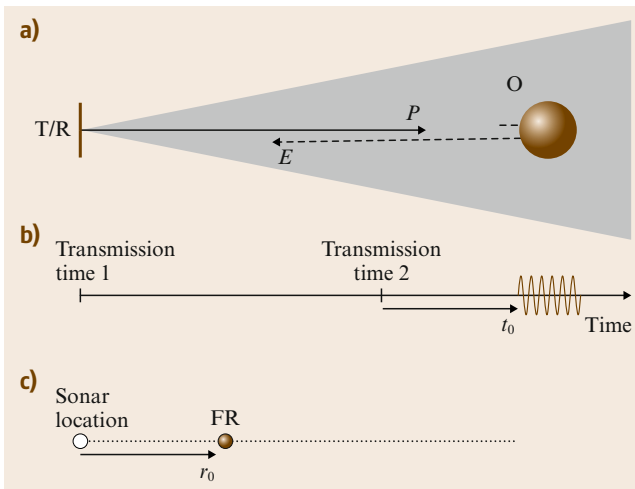


Fig. 21.2a–c False range reading: (a) sonar configuration, (b) probing pulse 2 transmitted before echo from pulse 1 arrives, (c) false range (FR) is measured from transmission time 2

Figure 21.1 shows a simplified sonar system, from its configuration to the resulting sonar map. A sonar transducer, T/R, acts as both the transmitter (T) of a probing acoustic pulse (P) and the receiver of echoes (E). An object O lying within the sonar beam, indicated by the shaded region, reflects the probing pulse. A part of the reflected signal impinges on the transducer as is detected as an echo. The echo travel time t_0 , commonly called the *time-of-flight* (TOF) is measured from the probing pulse transmission time. In this case the echo waveform is a replica of the probing pulse, which usually consists of as many as 16 cycles at the resonant frequency of the transducer. The object range r_0 is computed from t_0 using

$$r_0 = \frac{ct_0}{2}, \quad (21.1)$$

where c is the sound speed (343 m/s at standard temperature and pressure). The factor of 2 converts the round-trip (P+E) travel distance to a range measurement. Beam-spreading losses and acoustic absorption limit sonar range.

In forming a sonar map, a range dot is placed along the direction corresponding to the transducer's physical orientation. A sonar map is usually built by rotating the sensor about the vertical axis, indicated by the orientation angle θ , through a series of discrete angles separated by $\Delta\theta$ and placing sonar dots the corresponding ranges. Since the range from the object O to the center of T/R is almost constant as T/R rotates, the range dots typically fall on a circle as long as O lies within the beam. Hence, sonar maps are made up of arcs.

The major limitations of sonar include

1. the wide sonar beam causes poor directional resolution. Objects are located at the middle of isolated arcs, but shorter-range objects shorten the arcs of those at farther ranges, and the arcs produced by a collection of objects are often difficult to interpret. A consequence of this effect is that wide beams occlude small openings, limiting robot navigation,
2. the slow sound speed, relative to an optical sensor, reduces the sonar sensing rate. A new probing pulse should be transmitted after all detectable echoes from the previous pulse have expired, otherwise the false reading shown in Fig. 21.2 can occur. The echo from probing pulse 1 occurs after probing pulse 2 is emitted. Sonar measures the TOF from the most recent probing pulse. Many sonars transmit probing pulses every 50 ms, but encounter false readings in reverberant environments,
3. smooth surfaces at oblique incidence do not produce detectable echoes. Figure 21.3 shows a planar surface (a wall) that acts as a mirror to the sonar beam. The important point is that the nearby wall does not itself produce a detectable echo, and a robot using sonar for obstacle avoidance may collide with the wall,
4. artifacts caused by beam side-lobes and multiple reflections produce range readings in the environment where no objects exist. Figure 21.3 also shows the redirected beam enclosing object O. The echo is also redirected by the wall back to the transducer. From the transducer's reference, the object is at the virtual object location VO, and it would generate the same sonar map as shown in Fig. 21.1. Since there is no physical object corresponding to the sonar dot location, it is an artifact. Also, note that the acoustic energy indicated by the dot-dashed line reflected back to the transducer is not detected because it does not lie within the beam cone. Beam side-lobes often detect these echoes and produce

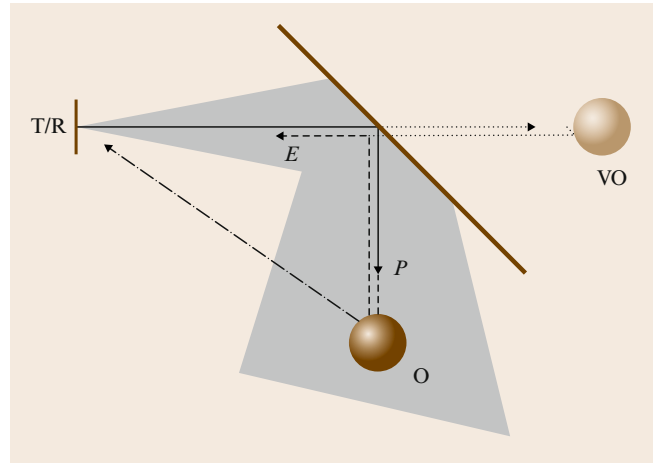


Fig. 21.3 The smooth surface redirects the beam, causing a sonar artifact at virtual object (VO) location. The dot-dashed echo path falls outside the sonar beam and does not produce a detectable echo

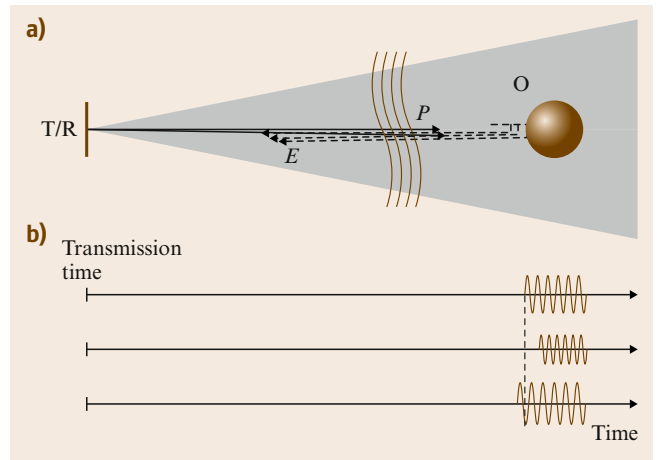


Fig. 21.4a,b Random echo jitter. (a) Sonar configuration; thermal inhomogeneities in the acoustic transmission medium cause refraction effects. (b) Examples of variations in echo travel times and amplitudes in a static environment

shorter-range readings but placed along the sonar orientation,

5. travel time and amplitude variations in the echoes caused by inhomogeneities in the sound speed. Both effects cause random fluctuations in the detected echo travel time, even in static environments. Figure 21.4 illustrates thermal fluctuations that cause speed up, retardation, and travel redirection by refraction of echoes. These cause temporal and amplitude variations in the echoes and jitter in the

range readings. While these typically introduce minor changes in sonar maps, they often cause havoc with approaches using finer analysis.

This chapter describes the physical and mathematical details that extend this simplified sonar model to practical sonar systems.

21.2 Sonar Beam Pattern

To derive a qualitative description of the sonar transducer, we apply elementary acoustics theory to a simplified model to achieve a simple analytic form [21.1]. A sonar emitter is commonly modeled as a circular piston surface of radius a vibrating with frequency f in an infinite planar baffle. The wavelength λ is

$$\lambda = \frac{c}{f}, \quad (21.2)$$

where c the sound speed in air, 343 m/s at 25 °C [21.2]. When $a > \lambda$ the emitted pressure field forms a beam consisting of a main lobe surrounded by side-lobes. In the far field, or range greater than a^2/λ , the beam is described by its *directivity pattern*, which is the two-dimensional Fourier transform of the aperture function, in this case the circular aperture produces a Bessel function. The emitted pressure amplitude at range r and angle θ relative to the piston axis can be written as

$$P_E(r, \theta) = \frac{\alpha a^2 f}{r} \left(\frac{2J_1(ka \sin \theta)}{ka \sin \theta} \right), \quad (21.3)$$

where α is a proportionality constant that includes the density of air and the source strength, $k = 2\pi/\lambda$, and J_1 is the Bessel function of the first kind. The term in the brackets evaluates to 1 along the sonar axis, $\theta = 0$. The a^2 term indicates that the emitted pressure increases with the piston area. The frequency f appears in the numerator because the faster-moving piston generates higher pressures. The range r appears in the denominator because the conservation of energy requires the pressure to decrease as the beam widens with range.

The main lobe is defined by its first off-axis null, which occurs at an angle of

$$\theta_0 = \arcsin \left(\frac{0.61\lambda}{a} \right) = 14.7^\circ. \quad (21.4)$$

For example, the popular electrostatic instrument grade transducer, formerly produced by Polaroid [21.3],

had a radius of $a = 1.8$ cm and is conventionally driven at $f = 49.4$ kHz, making $\lambda = 0.7$ cm and $\theta_0 = 14.7^\circ$.

An object that is small compared to λ and located in the emitted pressure field produces an echo with a spherical wavefront whose amplitude decays with the inverse

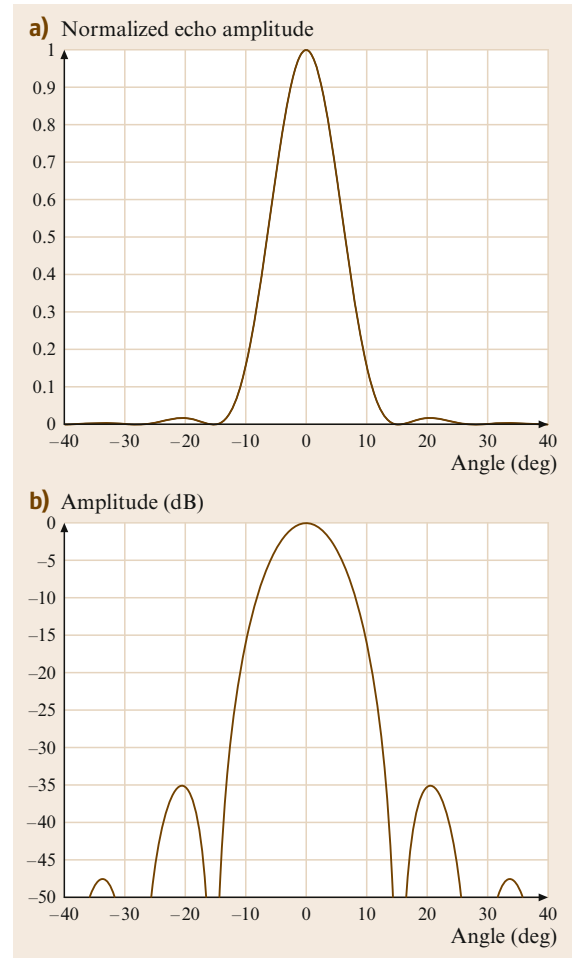


Fig. 21.5a,b Normalized amplitude of echo from small object predicted by the piston model as a function of angle. (a) Linear scale. (b) Decibel scale

of the distance propagated. In the common pulse-echo single transducer (*monostatic*) ranging sensor, only part of the echo wavefront impinges on the receiving aperture. The *sensitivity pattern* of the circular aperture, now acting as the receiver, has the same beam-like Bessel function form given in (21.3) by the reciprocity theorem [21.1]. If the reflecting object is located at (r, θ) relative to the transducer, the detected echo pressure amplitude, referenced to the receiver output, is given by

$$P_D(r, \theta) = \frac{\beta f a^4}{r^2} \left(\frac{2J_1(ka \sin \theta)}{ka \sin \theta} \right)^2, \quad (21.5)$$

where β is a proportionality constant that includes parameters that cannot be controlled in a design, such as the density of air. The additional a^2 in the numerator occurs because larger apertures detect more of the echo wavefront.

Figure 21.5 shows the echo amplitude from a small (point-like) object located in the far field as a function of angle detected by the electrostatic instrument grade transducer. The curve has been normalized by the on-axis echo amplitude.

This model is qualitative in that it provides the following practically useful insights:

- For a small reflector size relative to the wavelength, the echo amplitude decreases inversely with the square of the range because there is a $1/r$ dispersion loss from the transmitter to the object, followed by an additional $1/r$ dispersion loss in the echo back to the receiver. However, larger reflectors can be treated using a Huygens principle approach [21.4] by dividing them into smaller reflectors and coherently adding their echo contributions. When this is done in two dimensions over a normally incident extended plane reflector, the echo amplitude decreases as $1/r$ rather than $1/r^2$. A cylindrical reflector extends over one dimension and results in an amplitude variation with range between $1/r$ and $1/r^2$. A more-extreme situation can occur with a concave reflector that acts as an acoustic magnifier, resulting in an amplitude that decreases with a negative power of range of less than one.
- The transducer excited by an approximation to a sinusoid exhibits side-lobes due to null caused by phase cancellation. For example, the 16-cycle excitation employed in the conventional sonar exhibits side-lobes. The peak of the first side-lobe is -35 dB relative to the echo amplitude when a small reflec-

tor lies on the transducer axis. The specification sheet for the 600 series instrument-grade transducer shows the first off-axis null at 15° and a first side-lobe peak magnitude of -26 dB. We presume that these measurements were made using a plane as a reflector.

- This model can be used to compute approximate beam parameter values for other common transducers. For example, the SensComp 7000 Series [21.5] with $a = 1.25$ cm yields $\theta = 20^\circ$, equal to the specified value. However, the specified first side-lobe peak magnitude equals approximately -16 dB, which is substantially different from the expected -35 dB.

The limitations of the qualitative model include:

- Actual transducers only approximate pistons vibrating in an infinite planar baffle. The infinite baffle directs all the radiated sound pressure into the half-space in front of the transducer. Actual transducers radiate in all directions, but most of the acoustic energy is concentrated within the main lobe.
- All pulse-echo ranging sonars operate with finite-duration pulses rather than infinite-duration sinusoids. Several systems described below use pulses that are quite different from a sinusoidal excitation, either in duration or in form. These are commonly analyzed by computing the spectrum of the pulse and decomposing it into several sinusoidal frequencies, each having its own beam pattern. For example, the echo amplitude predictions above are reasonably accurate, including beam width and side-lobes, for the 16-cycle pulses. However, when impulse or swept-frequency excitations are used, the net beam profile becomes the superposition (of linear amplitudes) of the beam patterns produced by each frequency component in the excitation. Such broadband excitations do not exhibit nulls because the nulls formed by one frequency are filled in by main and side-lobes of beams produced by other frequencies.
- Most sonar transducers are encased in protective housings. The electrostatic instrument-grade transducer cover forms a mechanical filter that enhances the acoustic output at 49.4 kHz. The cases of other transducers may distort the transmitted field, but most form some type of directional beam.
- The model does not include frequency-dependent acoustic absorption of the transmission medium. These reduce the echo amplitudes predicted by the model.

The analytic model above is limited to simple configurations. With current computational power, transducers can be extended to those with arbitrary, even multiple, apertures and with various excitations. Waveforms of echoes from objects having arbitrary shapes can be simulated by using Huygens principle [21.4]. The transmitter, receiver, and object surfaces are broken up into two-dimensional surface arrays of emitting, reflecting, and detecting elements, using squares of dimension $< \lambda/5$ (the smaller the better, but taking longer). The impulse response of a given configuration is computed by as-

suming an impulsive emission and superimposing the travel times along all possible paths from all transmitter elements to all object elements and then to all receiver elements. The temporal resolution should be $< (20f_{\max})^{-1}$, where f_{\max} is the maximum frequency in the excitation. A $1\text{ }\mu\text{s}$ resolution is adequate for a 16-cycle 49.4 kHz excitation. A much finer resolution ($< 0.1\text{ }\mu\text{s}$) is required for an impulsive excitation. The echo waveform is then computed as the convolution of this impulse response with the actual transmitted pulse waveform [21.4].

21.3 Speed of Sound

The speed of sound c varies significantly with atmospheric temperature, pressure, and humidity and can be critical in determining the accuracy of a sonar system. This section outlines the relationship between c and these variables and is based on [21.6, 7].

The speed of sound in dry air at sea-level air density and one atmosphere pressure is given by

$$c_T = 20.05\sqrt{T_C + 273.16} \text{ ms}^{-1}, \quad (21.6)$$

where T_C is the temperature in degrees Celsius. Under most conditions (21.6) is accurate to within 1%. However, should the relative humidity be known, a better estimate can be made as

$$c_H = c_T + h_r [1.0059 \times 10^{-3} + 1.7776 \times 10^{-7} (T_C + 17.78)^3] \text{ ms}^{-1}. \quad (21.7)$$

Equation (21.7) is accurate to within 0.1% for temperatures in the range -30 – 43°C for most pressures at sea level. Should atmospheric pressure, p_s be known then

the following expression can be used

$$c_P = 20.05 \sqrt{\frac{T_C + 273.16}{1 - 3.79 \times 10^{-3} (h_r p_{\text{sat}} / p_s)}} \text{ ms}^{-1}, \quad (21.8)$$

where the saturation pressure of air, p_{sat} is dependent on temperature as follows

$$\begin{aligned} & \log_{10} \left(\frac{p_{\text{sat}}}{p_{s0}} \right) \\ &= 10.796 \left[1 - \left(\frac{T_{01}}{T} \right) \right] - 5.0281 \log_{10} \left(\frac{T_{01}}{T} \right) \\ & \quad + 1.5047 \times 10^{-4} \{ 1 - 10^{-8.2927[(T/T_{01}) - 1]} \} \\ & \quad + 0.42873 \times 10^{-3} \{ -1 + 10^{4.7696[1 - (T_{01}/T)]} \} \\ & \quad - 2.2196, \end{aligned} \quad (21.9)$$

p_{s0} is the reference atmospheric pressure of 101.325 kPa and T_{01} is the triple-point isotherm temperature with the exact value of 273.16 K.

21.4 Waveforms

Sonars employ a variety of waveforms, the most common types of which are shown in Fig. 21.6. Each waveform can be considered the echo from a normally incident plane. Waveforms are classified as being narrow- or wide-band depending on their spectral bandwidth. Narrow-band pulses provide superior detection

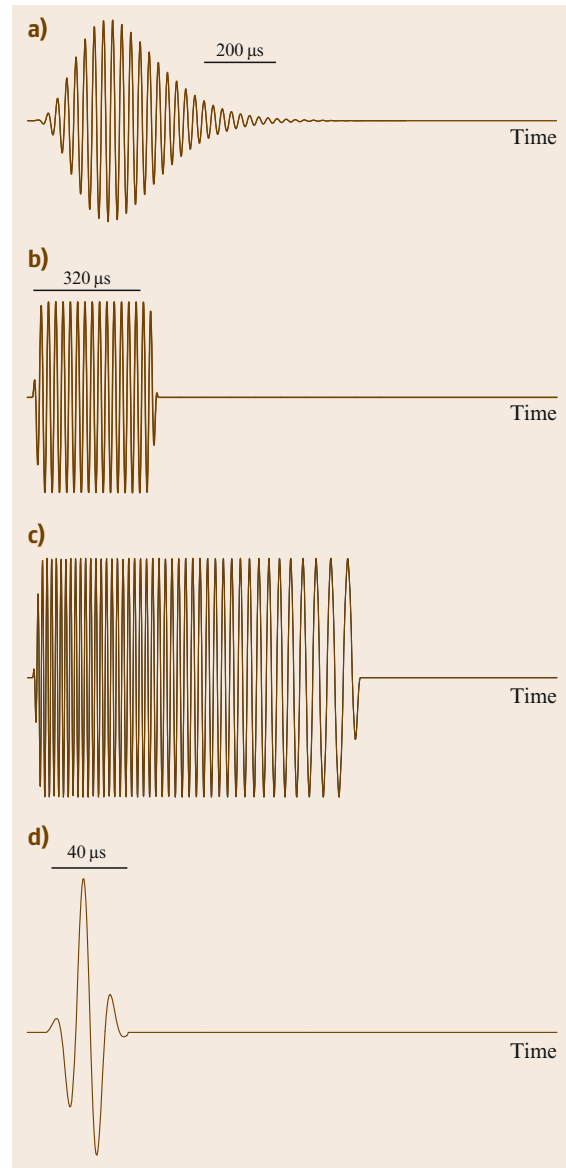
performance in the presence of additive noise, while wide-band pulses provide better range resolution and do not have side-lobes.

Figure 21.6a shows the waveform produced by the Murata 40 kHz piezoelectric transducer excited by an eight-cycle 40 kHz square wave 40 V_{rms} . The Murata

Fig. 21.6a–d Common sonar pulse waveforms. (a) Murata 40 kHz transducer (narrow band). (b) Polaroid 600 electrostatic transducer excited with 16-cycle sinusoid at 49.4 kHz (6500 ranging module – narrow band). (c) Polaroid 600 electrostatic transducer excited with decreasing-frequency excitation signal (wide band). (d) Polaroid 600 electrostatic transducer excited with 10 μ s 300 V pulse (wide band) ►

sensor is small, lightweight, and efficient, but has an approximately 90° beam width. These transducers are used in monostatic, bistatic, and multiple transducer arrays [21.8, 9].

The next three waveforms were produced by the Polaroid 600 electrostatic transducer. Similar waveforms are generated by the smaller Polaroid 7000 transducer. Figure 21.6b shows the waveform produced by the 6500 ranging module. This ranging module with its 10 m range, low cost, and simple digital interface is a popular choice for implementing sonar arrays and rings. While the electrostatic transducer is inherently wide-band, with a usable frequency range from 10 to 120 kHz [21.10], narrow-band pulses are produced by exciting the transducer with 16 cycles at 49.4 kHz. Figure 21.6c illustrates a means to exploit the wide bandwidth of the Polaroid electrostatic transducer by exciting it with a decreasing-frequency square wave. Such frequency-sweep pulses are processed by a band of band-pass filters to extract the frequency dependence of reflecting objects. A correlation detector, also known as a matched filter, compresses swept-frequency pulses to improve range resolution. Longer-duration (100 ms) pulses are used in CTFM systems. Figure 21.6d shows a wide-band pulse when the excitation is a 10 μ s-duration 300 V pulse. The metal protective mesh, which also acts as a mechanical filter resonant at 50 kHz, was removed by machining to achieve a usable bandwidth from 10 kHz to 120 kHz, with the peak occurring at 60 kHz. Such wide-band pulses are useful for object classification [21.10, 11]. These pulses have small amplitudes, limiting their range to 1 m or less.



21.5 Transducer Technologies

Electrostatic and piezoelectric transducers are the two major types available that operate in air and can in principle operate both as a transmitter and receiver – some samples are shown in Fig. 21.7. In general electrostatic devices have a higher sensitivity and bandwidth but typ-

ically require a bias voltage above 100 V. Piezoelectric devices operate at lower voltages, making their electronic interfacing simpler, but have a high- Q resonant ceramic crystal and this results in a narrow frequency response compared to electrostatic transducers.



Fig. 21.7 Left to right: Series 9000, instrument-grade, and Series 7000 transducers; front and back views are shown (Photo courtesy Acroname, Inc., Boulder; www.acroname.com)

21.5.1 Electrostatic

An example of an electrostatic transducer is the Polaroid instrument-grade transducer (now available from SensComp.com) constructed from a gold-coated plastic foil membrane stretched across a round grooved aluminium back plate. The conductive foil is charged via a bias voltage of 150 V with respect to the back plate. Incoming sound waves vibrate the foil and change the average distance between the foil and back plate and thereby changing the capacitance of the foil. Assuming that the charge q is constant, the voltage $v(t)$ is generated proportional to this varying capacitance $C(t)$ as $v(t) = qC(t)$. As a transmitter, the transducer membrane is vibrated by applying 0–300 V pulses across this capacitor, typically using a pulse transformer. The charge induced by the 300 V on the capacitor causes an electrostatic attraction force between the membrane and the back plate. The grooves on the back plate allow stretching of the membrane and by creating randomness in the back plate roughness a broad resonance can be achieved in the frequency response. For example the bandwidth of the 7000 series Polaroid transducer is 20 kHz. A front grille is mounted on the transducer and removing this grille reduces losses and reverberation between the grill and the membrane. Another electrostatic transducer was designed by Kay and details of its design can be found in [21.12].

21.5.2 Piezoelectric

Piezoelectric ceramic transducers can be used as both transmitters and receivers, however some manufacturers sell transmitters and receivers separately in order to optimize the transmitted power and receiver sensitivity,

respectively. A piezoelectric resonant crystal mechanically vibrates when a voltage is applied across the crystal, and in reverse generates a voltage when mechanically vibrated. Often a conical concave horn is mounted on the crystal to acoustically match the crystal acoustic impedance to that of air. An example is the Murata MA40A5R/S receiver and sender transducers, which operate at 40 kHz. This device has a diameter of 16 mm and a 60° beam angle for transmitter combined with receiver for –20 dB loss compared to the maximum sensitivity. The effective bandwidth of transmitter and receiver is only a few kHz due to the resonant nature of the crystals. This limits the envelope rise time of pulses to around 0.5 ms. An advantage is the ability to drive piezoelectric devices with low voltages, for example by connecting each terminal to complementary CMOS logic outputs. There is a wide range of resonant frequencies for piezoelectric transducers from 20 kHz to megahertz. Also available is piezoelectric film called polarized fluoropolymer, polyvinylidene fluoride (PVDF) from www.msusa.com. This flexible film can be cut to shape and custom ultrasonic transmitters and receivers can be formed. The sensitivities of the transmitters and receivers made from PVDF is generally lower than that of ceramic crystal transducers and most applications are short range where the broadband nature of PVDF allows short pulses to be formed, allowing pulse-echo ranging to as little as 30 mm.

21.5.3 MEMS

Microelectromechanical system (MEMS) ultrasonic transducers can be fabricated on a silicon chip and integrated with electronics. The sensors offer a low-cost mass-produced alternative to standard transducers. MEMS ultrasonic transducers operate as electrostatic capacitive transducers where the membrane can be made from thin nitride. Devices operate at frequencies up to several megahertz and offer advantages in signal-to-noise ratio over piezoelectric devices due to their better matching to air acoustic impedance [21.13]. Two-dimensional arrays of devices can be deployed on a chip that are well matched and steerable.

21.6 Reflecting Object Models

Modeling the reflection processes helps in interpreting echo information. In this section we consider three simple reflector models: planes, corners, and edges, shown in Fig. 21.8. These models apply to both single transducers and arrays.

A plane is a smooth surface that acts as an acoustic mirror. Smooth walls and door surfaces act as planar reflectors. The plane must be sufficiently wide to produce the two reflections whose path is shown in dotted line. The plane reflector is then slightly larger than the intersection area of the beam with a plane of infinite extent. Smaller planes produce weaker echoes because of a smaller reflecting surface and negative interference by echoes diffracted from the edges of the plane. An acoustic mirror allows the analysis using a virtual transducer, indicated by primes in the figure.

A corner is the concave right-angle intersection of two surfaces. Corners formed by intersecting walls, the sides of file cabinets, and door jambs are commonly observed corner reflectors in indoor environments. The novel feature of the corner, and its three-dimensional (3-D) counterpart the corner cube, is that waves reflect back in the same direction from which they originate. This is caused by planar reflections at each of the two surfaces defining the corner. The virtual transducer is then obtained by reflecting the transducer about one plane of the corner and then the other plane. This gives rise to a reflection through the intersection point of the corner as shown in Fig. 21.8b. The virtual transducer analysis indicates that, for a monostatic sonar, echoes from a plane and corner are identical and that planes and corners can generate identical sonar maps [21.4]. The difference in the virtual transducer orientation between planes and corners has been exploited using transducer arrays to differentiate these reflectors [21.11, 14].

The edge shown in Fig. 21.8c models physical objects such as convex corners and high-curvature surfaces (posts), where the point of reflection is approximately independent of transducer position. Edges are encountered in hallways. While planes and corners generate strong echoes, edges generate weak echoes that are detected only a short range [21.4], making them difficult objects to detect. Early robot sonar researchers placed bubble wrap material on edge surfaces to make them reliably detectable.

Many environmental objects can be configured as a collection of planes, corners, and edges. Models for echo production [21.15, 16] indicate that normally in-

cident surface patches and locations at which sharp changes in the surface function and its derivatives generate echoes. Objects with rough surfaces or a collection of many objects generate echoes from a variety of ranges and bearings, as illustrated in Fig. 21.9. If $p(t)$ repre-

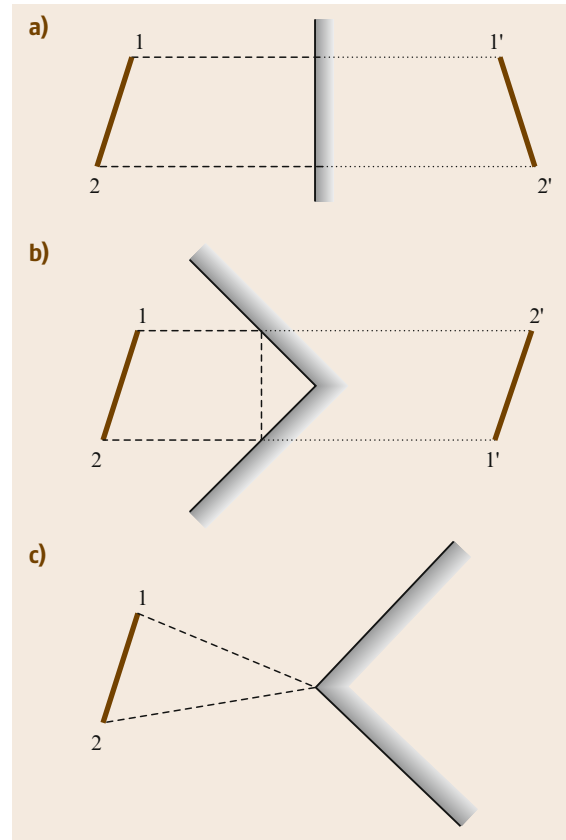


Fig. 21.8a–c Reflector models: (a) plane, (b) corner, and (c) edge

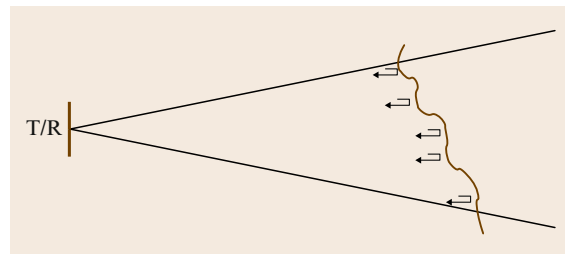


Fig. 21.9 Random reflector model. Echoes reflect back from normally incident surface sections within the beam

sents a single echo waveform, often a replica of the probing waveform, the total echo waveform $p_T(t)$ is the sum of individual echoes $p_i(t)$ from N normally incident patches at range r_i and bearing θ_i , scaled by amplitude a_i , or

$$p_T(t) = \sum_{i=1}^N a_i(\theta_i) p_i\left(t - \frac{2r_i}{c}\right), \quad (21.10)$$

where $a_i(\theta_i)$ is an amplitude factor related to the surface patch size and its bearing in the beam. Wide-bandwidth echoes are more complicated because their

waveform changes in a deterministic fashion due to diffraction [21.11].

Sonars that analyze $p_T(t)$ employ analog-to-digital converters to obtain waveform samples [21.11, 17]. Reflecting patches separated in range produce isolated patches [21.11], but more often the incremental travel time is less than the pulse duration, causing pulse overlap. Rough surfaces and volume scatterers, such as indoor foliage, have large N , allowing $p_T(t)$ to be treated as a random process [21.18, 19]. Conventional TOF sonars output the first time that $p_T(t)$ exceeds a threshold [21.11].

21.7 Artifacts

Sonars usually work well in simple environments, while complex environments often produce mysterious readings, artifacts, that foil attempts to build reliable sonar maps. Artifacts have given sonar a bad reputation as being a noisy, or low-quality, sensing modality. Sonar stalwarts believe sonar would open up many new applications, *if only* we understood echoes at a level that approximates that employed by bats and dolphins [21.20]. Sonar stalwarts divide into two categories in terms of how they treat artifacts. The first attempts to build *intelligent sensors* that identify and suppresses artifacts before transmitting data to a higher-level reasoning program. Previous approaches [21.21, 22] required custom electronics, which other researchers have been reluctant to adopt because of expense or lack of experience. An alternate approach is to control conventional sonar in a novel fashion to produce a series of spikes and requires only software changes [21.23]. Sonar arrays have been used to find consistent data [21.24–26]. Echoes from specular reflectors, such as planes, corners, or posts, exhibit detectable features, which can be obscured by artifacts.

The second category of sonar users attempts to eliminate artifacts produced by conventional sensors by using higher-level postprocessing. These include proponents of occupancy (or certainty) grids [21.27, 28], including those that apply simplified physical models, such as sonar arcs [21.29, 30]. In simple environments postprocessing usually eliminates artifacts that are inconsistent with a feature [21.31] or with a physical map [21.30]. More-sophisticated methods handle artifacts by treating them as noise and applying hidden Markov models (HMM) [21.32]. However, multiple passes are needed to successfully teach the system about relatively simple environments, mostly because artifacts are not amenable

to being treated as independent additive noise. Eliminating troublesome artifacts would replace HMM with simpler Markov chains [21.33, 34], and sufficient sonar data can be obtained in a single pass. What frustrates this second category, and mildly amuses the first, is that this postprocessing works well *in simple environments*, but fails in real-world environments. This second category eventually abandons sonar and joins the camera and laser ranging crowd.

There are two important classes of artifacts: axial multiple reflection (MR) artifacts and dynamic artifacts. These artifacts are important in sonar mapping when they indicate the presence of a static object at a location where none exists. Troublesome MR artifacts are caused by delayed echoes produced by a previous probing pulse exceeding the detection threshold after the current probing pulse has been transmitted. Such artifacts then appear as close-range objects and obscure actual farther-range objects in conventional sonars. Most sonars employ probing pulse emission periods longer than 50 ms to avoid MR artifacts, although some reverberant environments can still produce artifacts [21.35].

Dynamic artifacts are produced by moving objects, such as individuals passing through the sonar beam. Even though these are actual objects and echoes indicate their true range, their presence should not be part of a sonar map that describes the static environment. Such dynamic artifacts make quantitative matchings between stored and generated sonar maps error-prone.

Another common artifact is a nonaxial MR artifact [21.4] caused by an obliquely incident smooth surface that redirects the sonar beam to some other echo-producing object. The TOF produces a range reading that is positioned along the sonar axis. While the object is

not at the location indicated on a sonar map, its location in the sonar map is a stable element and can be useful for navigation.

One may argue that, if the locations of all objects are known, the echoes can be determined and should not be treated as random processes. However, the presence of speed fluctuations in the medium due to thermal gradients and ever-present electronic noise cause random fluctuations in the times that thresholds are exceeded. Even a stationary sonar in a static environment exhibits random fluctuations [21.36], similar to the visual experience of fading when viewing objects beyond a heated surface.

21.8 TOF Ranging

Most conventional sonars employ Polaroid 6500 ranging modules [21.37] connected to the Polaroid 600 series electrostatic ultrasound transducer. The module is controlled with digital signals on two input lines (*INIT* for initialization and probing pulse transmission and *BLNK* for clearing the indication and resetting the detector) and the *TOF* reading occurs on its output line (*ECHO*). A logic transition on *INIT* causes the transducer to emit a pulse lasting for 16 cycles at 49.4 kHz. The same transducer detects echoes after a short delay to allow transmission transients to decay. Another interrogation pulse is typically emitted only after all the

Sonar can identify artifacts by applying three physical criteria that are met by echoes from static environmental objects. Artifact features include [21.35]

1. echo amplitude – echoes with amplitudes less than a specified threshold;
2. coherence – echoes forming constant-range azimuthal intervals less than a specified threshold; and
3. coincidence – echoes detected with a sonar array at different times (lacking temporal coherence) or corresponding to different locations (lacking spatial coherence).

echoes produced by the previous pulse have decayed below a detection threshold.

The module processes echoes by performing rectification and lossy integration. Figure 21.10 illustrates a simulation of the processed waveform applied to the threshold detector. While the echo arrives at time t_0 after the emission, *ECHO* exhibits a transition at the measured *TOF* time t_m , the first time the processed echo signal exceeds a detection threshold τ . By convention, the range r of the reflecting object is calculated by

$$r = \frac{ct_m}{2}, \quad (21.11)$$

where c is the speed of sound in air, usually taken as 343 m/s.

Figure 21.10b shows details around the threshold detection point including the residual high-frequency ripple after full-wave rectification and integration. Two effects can be noticed. First, t_m will always occur after t_0 , making threshold detection a biased estimate of the true echo arrival time. Moreover, this bias is related to the echo amplitude: stronger echoes will produce an integrator output having a greater slope, which exceeds τ sooner than t_0 . Second, as the echo amplitude decreases, for example, when the object moves away from the transducer axis, the threshold level occurs later in the integrator output and t_m will experience small jumps in time approximately equal to half the period [21.38].

The first step in developing a model of the detection process is to develop a model for the echo amplitude as a function of bearing. The Polaroid transducer is often modeled as a vibrating piston to yield the transmitter/receiver beam pattern shown in Fig. 21.11. To

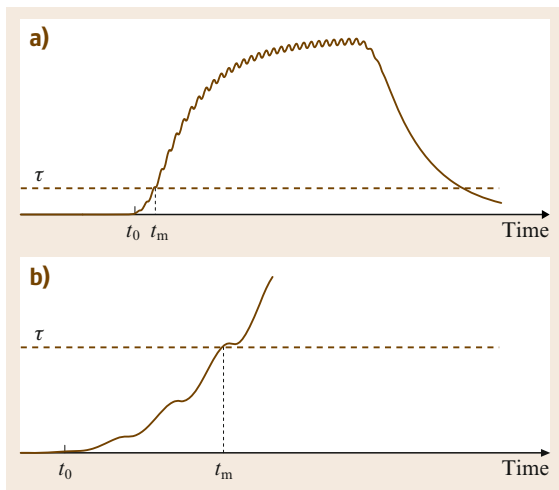


Fig. 21.10a,b Simulation of Polaroid ranging module operation. (a) Processed echo waveform. (b) Expanded time and amplitude scale around threshold crossing point

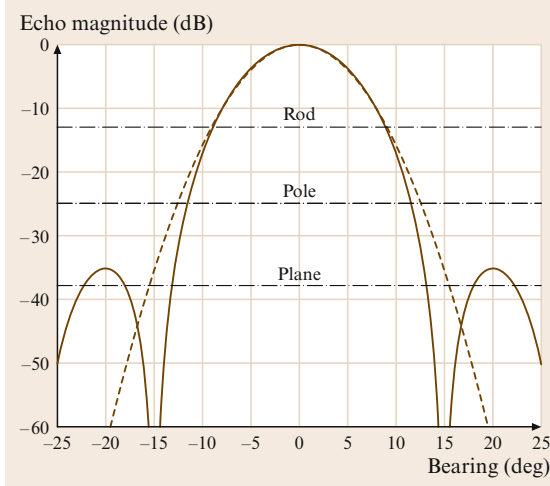


Fig. 21.11 Piston model transmitter/receiver pattern for the Polaroid 600 series transducer. A Gaussian approximation with $SD = 5.25^\circ$ is shown in *dashed line*. Equivalent threshold levels for plane, pole, and rod objects at 1.5 m range are shown as *dot-dashed lines*

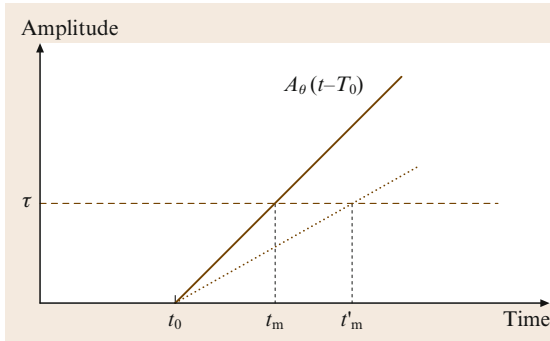
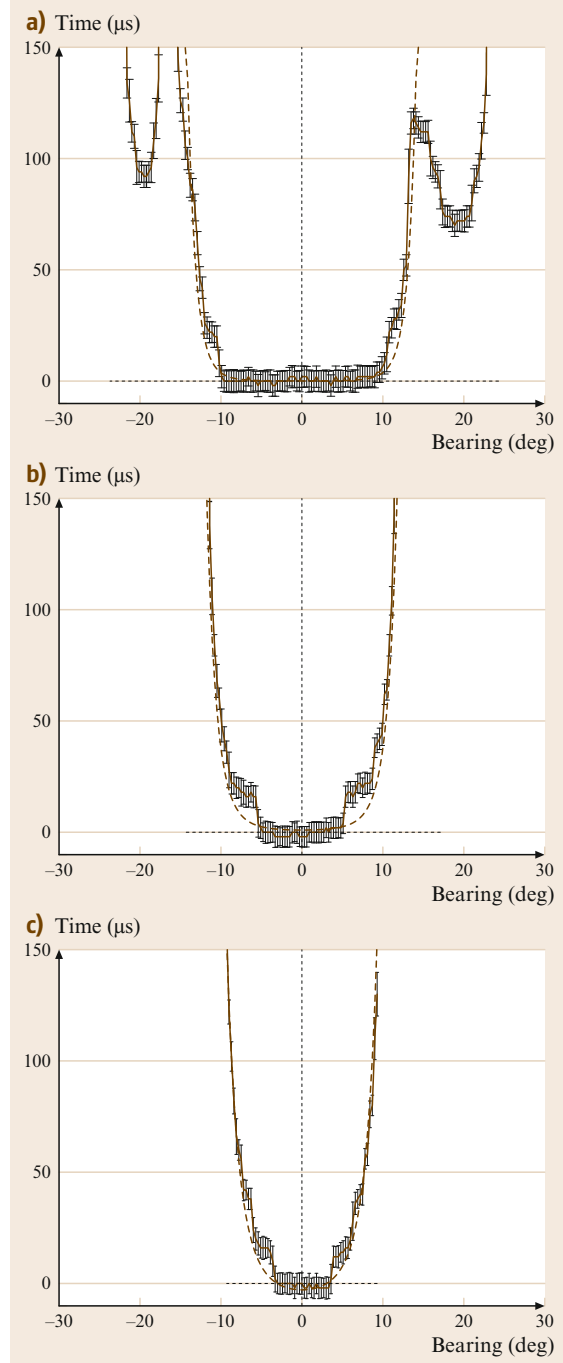


Fig. 21.12 TOF values t_m and t'_m for idealized processed echo waveforms having two amplitudes. The *solid line* indicates a larger-amplitude echo

Fig. 21.13a–c TOF data from an object at 1.5 m range; mean of 100 measurements with bars indicating ± 1 SD. The *dashed lines* are model predictions. **(a)** 1 m wide plane ($\tau/A = 0.15 \mu s$). **(b)** 8.9 cm diam pole ($\tau/A = 0.67 \mu s$). **(c)** 8 mm diam rod ($\tau/A = 2.68 \mu s$) ►

simplify the analysis, the peak of the beam profile is approximated with a Gaussian function, a parabola in logarithmic units in Fig. 21.11, to determine the echo amplitude as a function of object bearing θ , or

$$A_\theta = A_0 \exp\left(-\frac{\theta^2}{2\sigma^2}\right), \quad (21.12)$$



where A_0 is the on-axis amplitude and σ is a measure of the beam width. The value $\sigma = 5.25^\circ$ provides a good fit around the peak of the beam pattern. The

Gaussian model is reasonable only over the central section of the main lobe that produces detectable echoes.

We assume that the echo arrival time t_0 does not change significantly with transducer orientation (object bearing); this effect was investigated and found to be minor [21.4]. In contrast, measured TOFs, denoted by t_m and t'_m in Fig. 21.12, are amplitude dependent and a function of object bearing, which affects echo amplitude as shown in Fig. 21.11.

The module processes the detected echo waveform by rectification and lossy integration, as discussed above. To derive a useful analytic model, assume that the integration is lossless and the rectified echo is a unit step function with amplitude A . This approximates the processed waveform shown in Fig. 21.10b by a linear function around time t_m , shown in Fig. 21.12. The model ignores the residual ripple and the decreasing slope of the waveform as the lossy rectification approaches a constant value, shown in Fig. 21.10. The linear function with a slope proportional to the echo amplitude is given by $A_\theta(t - t_0)$, for $t \geq t_0$. This function exceeds the threshold τ at

$$t_m = t_0 + \frac{\tau}{A_\theta} = t_0 + \frac{\tau}{A_0} \exp\left(\frac{\theta^2}{2\sigma^2}\right). \quad (21.13)$$

For fixed τ , the incremental delay in t_m is a function of the bearing θ and inversely proportional to the echo amplitude. When a constant echo amplitude A (in volts) is applied to the integrator, the slope of the linear output is A V/s, with typical values on the order of $A_\theta = 10^5$ V/s. If $\tau = 0.10$ V, $\tau/A_\theta = 10^{-6}$ s = 1 μ s.

Experiments were conducted with a Polaroid 600 series transducer connected to a model 6500 ranging module [21.38]. The Polaroid module was operated conventionally to generate t_m values as a rotational scan was performed. Objects include a 1 m-wide plane, a 8.9 cm-diameter pole, and an 8 mm-diameter rod, all located at 1.5 m range. A rotational scan was performed from -40° to $+40^\circ$ in 0.3° steps. At each angle, 100 t_m values were recorded. The mean deviations from the t_m when the object is on the sonar axis ($\theta = 0$) were determined

and the standard deviation (SD) values were computed. There were no other objects in proximity to the object being scanned. Echoes from objects beyond 2 m were eliminated by a range gate.

Figure 21.13a shows the data for the plane, Fig. 21.13b for the pole, and Fig. 21.13c for the rod. The values are shown relative to the t_m value observed at 0° bearing. Dashed lines indicate the values predicted by the model. The t_m values showed a variation with $SD = 5 \mu$ s (0.9 mm) at zero bearing, which is about nine times greater than that predicted by sampling jitter alone. This random time jitter is caused by dynamic thermal inhomogeneities in the air transmission medium, which change the local sound speed and cause refraction [21.36, 39]. The SD increases with deviation from zero bearing because smaller echoes exceed the threshold later in the processed waveform. The smaller slope of the latter part of the processed echo waveform shown in Fig. 21.10 causes greater t_m differences for a given variation in echo amplitude, thus increasing the SD.

One feature in the data not described by the model is due to residual ripple in the integrator output, which causes jumps in TOF readings equal to half periods (10 μ s) added to the value predicted by (21.13). These jumps are clearly evident in the mean values of Fig. 21.13.

The angular extent over which echoes were detected was 45° for the plane, 22.8° for the pole, and 18.6° for the rod. Side-lobes produced by the plane are visible and have small echo amplitudes, which cause their t_m values to be retarded in time. These angular extents can be related to the echo amplitudes that would have produced the respective arcs according to the piston model; these are indicated in Fig. 21.11. For the plane, the threshold level relative to the maximum echo amplitude is -38 dB, for the pole -25 dB, and for the rod -13 dB. Since the ranging module threshold at 1.5 m range is the same for each object, the difference in levels indicates the relative echo strength from each object, i.e., the plane echo is 13 dB (a factor of 4.5) greater than the pole echo, and the pole echo is 12 dB (a factor of 4) greater than the rod echo.

21.9 Echo Waveform Coding

Systems that display echo information beyond the first echo have been investigated [21.11, 17, 24, 40–42], but typically employ custom electronics. One motivation for examining the entire echo waveform is the success of

diagnostic medical ultrasound imaging systems, which adopt this method [21.43, 44].

As a less expensive alternative to analog-to-digital conversion, the Polaroid ranging module can detect

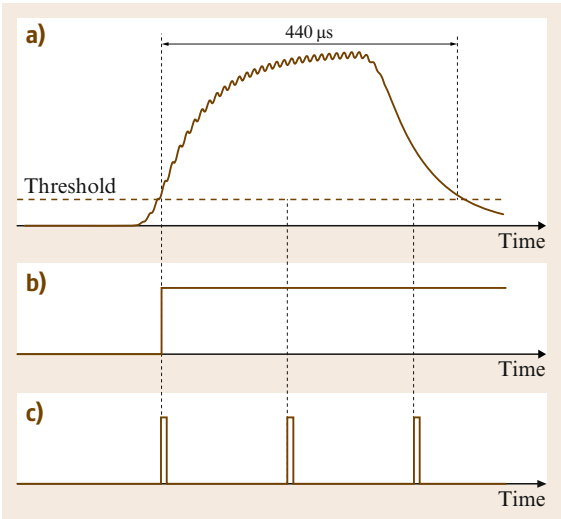


Fig. 21.14a–c Polaroid ranging module operation modes. (a) Processed echo waveform. (b) *ECHO* output produced in conventional time-of-flight mode. (c) *ECHO* output produced in *PAS* mode

echoes beyond the initial echo by repeatedly resetting the detection circuit. The 6500 module specification suggests a delay before resetting to prevent the current echo from retriggering the detection circuit [21.3]. Let us ignore this suggestion and control the Polaroid module in nonstandard way to provide information about the entire echo waveform. Since the echo amplitude is estimated from the digital output produced by the Polaroid module, this operation has been called *pseudo-amplitude scan (PAS)* sonar [21.23].

The conventional ranging module processes detected echoes by performing rectification and forming a lossy integration, as illustrated in Fig. 21.14a.

The *BLNK* input is typically kept at zero logic level, which enables the *ECHO* output. *ECHO* exhibits a transition at the time when the processed echo signal exceeds a threshold, as shown in Fig. 21.14b. By convention, the time interval between the *INIT* and *ECHO* transitions indicates the time of flight (*TOF*), from which the range *r* of the reflecting object is calculated by

$$r = \frac{c \times \text{TOF}}{2} . \tag{21.14}$$

Echoes occurring after the initial echo can be detected by resetting *ECHO* by pulsing the *BLNK* input. The specification suggests that the *BLNK* pulse should be delayed after the *ECHO* indication by at least 440 μs to account for all 16 returning cycles in the echo and to

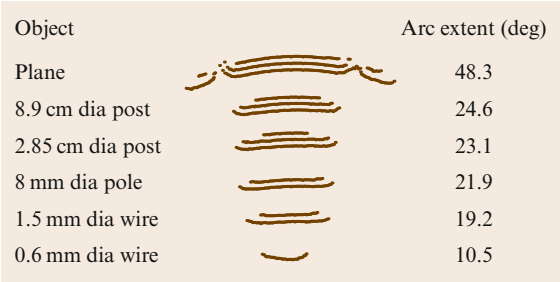


Fig. 21.15 *PAS* sonar maps of six objects located at 1 m range. The sonar is located below the objects in the figure

allow it to decay below the threshold for the largest observable echo. The largest echoes typically saturate the detection circuit, providing a predetermined maximum value. This duration corresponds to the time interval over which the processed signal is above the threshold, as shown in Fig. 21.14a.

When an *ECHO* event is observed, the *PAS* system issues a short 3 μs (corresponding to a software query period) pulse on the *BLNK* input line, which clears the *ECHO* signal as shown in Fig. 21.14c. Upon being cleared, the Polaroid module exhibits a delay inversely related to the echo amplitude, lasting at least 140 μs for large-amplitude echoes, and then produces another *ECHO* event if the processed echo signal still exceeds the threshold. The *PAS* system repeatedly issues a *BLNK* pulse whenever an *ECHO* event is observed. Hence, a strong echo is represented by three pulses on the *ECHO* line, the first corresponding to the conventional *TOF*, followed by two more pulses. Because lower amplitude echoes spend less time above the threshold, a weaker echo produces two pulses spaced farther apart, and a very weak echo may produce only one pulse.

A *PAS* sonar map is generated by placing a range dot along the transducer axis as a rotational scan is executed. With multiple readings per interrogation pulse, a *PAS* sonar map contains multiple dots at each interrogation angle. Rotational scans then form arcs, with isolated arcs indicating weak echoes, arc pairs moderate echoes, and arc triplets large echoes. To illustrate, Fig. 21.15 shows arcs formed by a large plane (2.3 m width by 0.6 m height) and five cylinders with different diameters, all placed at 1 m range. Examining objects at the same range eliminates effects caused by the range-dependent gain of the module.

A conventional *TOF* sonar map by comparison would display only the nearest arc in the *PAS* map for each object. Qualitatively, the arc length increases and the number of arcs increase with the echo amplitude,

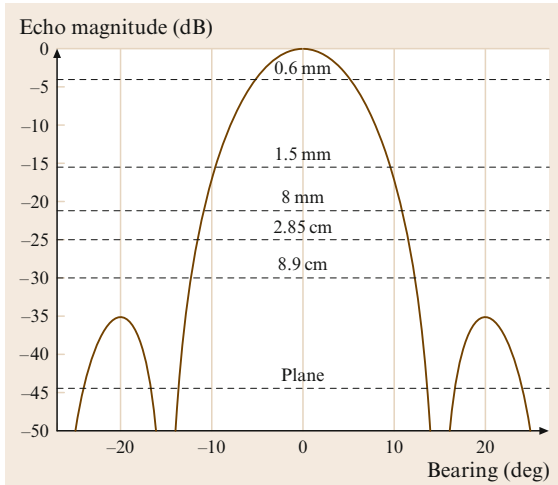


Fig. 21.16 Transmitter-receiver beam pattern. Dashed lines indicate the equivalent threshold level for each object

which is bearing dependent. The strongest reflectors produce concave arcs [21.4,45]. This occurs because, with echo amplitudes much greater than the threshold, the threshold is exceeded near the beginning of the echo, yielding a nearly constant range reading over a significant extent over bearing. In contrast, the weakest reflectors produce convex arcs, caused by echoes whose amplitudes are comparable to the threshold. As the echo amplitude decreases the threshold is exceeded at later points along the processed waveform, producing greater range readings. This effect also appears at the edges of the arcs produced by strong reflectors.

Computing the beam pattern of the vibrating piston model [21.1], which is a reasonable approximation to the Polaroid transducer, yields the curve shown in Fig. 21.16. This figure describes the detected echo magnitude normalized to have a maximum of 0 dB, which occurs along the beam axis. The larger echoes are much greater than the threshold, such as those produced by the plane, whose maximum amplitudes can be 44 dB relative to the threshold. The -44 dB threshold agrees with the PAS map for a plane: strong echoes (three stripes) occur within 10° of normal incidence, range readings increase due to echo amplitude reduction at $\pm 15.6^\circ$, approximating the predicted nulls at $\pm 14.7^\circ$, and smaller-amplitude echoes from the side-lobes are present. Weaker reflectors correspond to larger thresholds when their on-axis echoes normalize to 0 dB. The beam pattern model explains how arc length varies with object reflecting strength. The indicated thresholds were found by matching the angular beam width to the arc extent.

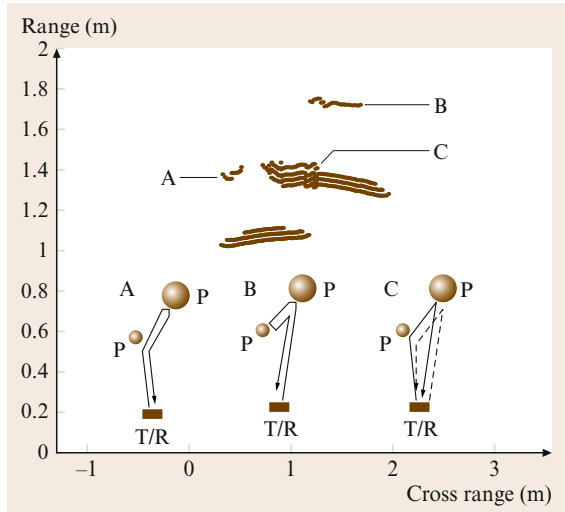


Fig. 21.17 PAS sonar map of 2.85 cm-diameter post (p) and 8.9 cm-diameter post (P). The transducer is located at (0,0). A: Artifact caused by echo originating at transmitter T, reflected by p, bouncing off P back to p, and directed toward receiver R. ($T \rightarrow p \rightarrow P \rightarrow p \rightarrow R$). B: $T \rightarrow P \rightarrow p \rightarrow P \rightarrow R$. C: $T \rightarrow p \rightarrow P \rightarrow R$ and $T \rightarrow P \rightarrow p \rightarrow R$

It is apparent that PAS maps provide information useful for solving the inverse problem, that of determining the identity of the object from the echoes. Figure 21.15 shows that PAS maps contain information about the echo amplitude. While it is true that the conventional TOF sonar maps, represented by the closest arc, can determine the object location from the arc center and can infer the echo amplitude from the arc extent, it is also true that this information is presented in a more robust way in the PAS maps. For this simple case of isolated objects, the posts can be clearly differentiated from the pole, while the corresponding conventional TOF arcs are comparable. A tenfold increase in post diameter yields only a modest increase in the conventional TOF arc length, while increasing the number of arcs from two to three in the PAS maps.

When examining the entire echo waveform, one must account for artifacts that are produced when objects interact acoustically. Some artifacts occur after the first detected echo, so these are not a problem in TOF sonar maps [21.4], but must be addressed in interpreting PAS maps. Consider a simple environment consisting of two posts: a 2.85 cm-diameter post (p), located at $r = 1$ m and bearing 12° , and an 8.9 cm-diameter post (P), at $r = 1.3$ m and bearing -10° . The corresponding

PAS map shown in Fig. 21.17 displays the echoes from the two objects plus additional echoes that illustrate two types of multiple reflection artifacts. The first type, indicated by A and B, results when only one object is within the transducer beam. An interrogation pulse that is redirected by a reflected object must be directed back to the receiver within its beam pattern in order to be detected. The paths that do this are shown in the figure. The single-arc convex shape of A indicates that the echo has a small amplitude. This is reasonable since both reflectors are nonplanar, and hence weak.

The second type of artifact (C) shown in Fig. 21.17 occurs when both objects are within the beam pattern.

This allows two distinct paths for the echoes to return to the receiver, occurring in opposite directions and doubling the artifact amplitude. With both objects lying near the beam edges, the echo amplitude is small. Since the distance traveled by these echoes is slightly greater than the range to the farther object, this artifact shows a range slightly beyond the more distant object. The superposition of these two components makes the echo from the farther object appear spread out in time. This pulse stretching explains why four arcs are observed, and at one angle five arcs. If the bearing angle between p and P was increased to exceed the beam width, this artifact would disappear.

21.10 Echo Waveform Processing

In this section pulse-echo sonar that processes sampled digitized receiver waveforms is described. These systems offer superior performance over the simple Polaroid ranging module systems described above, which report the TOF based on a threshold. Echo waveform processing does however incur the overhead of more-complex electronics and signal processing and is not readily available commercially.

21.10.1 Ranging and Wide-Bandwidth Pulses

It is shown in [21.11, 46] that the maximum-likelihood estimator (MLE) for the TOF is obtained by maximizing the correlation $\text{cor}(\tau)$ between the received pulse $p(t)$ (containing Gaussian white noise) and the known pulse shape shifted by τ , $\text{rec}(t - \tau)$

$$\text{cor}(\tau) = \frac{\int_a^b p(t) \text{rec}(t - \tau) dt}{\sqrt{\int_a^b p^2(t) dt \int_a^b \text{rec}^2(t) dt}}, \quad (21.15)$$

where the pulse extends from time a to b . The known pulse shape at the receiver depends on the angle of transmission and reception with respect to the normals of the respective transducers. The pulse shape can be obtained by collecting a good signal to noise pulse at 1 m range at normal incidence to the receiver and transmitter and using elliptical impulse response models to obtain template pulses at angles different to normal incidence. Pulse shape also changes with range due to the dispersive properties of absorption due losses in air transmission. These can be modeled using an estimate of the impulse

response due to one meter path through air as is done in [21.11].

The correlation, $\text{cor}(\tau)$ is normalized in (21.15) to be between -1 and $+1$. The correlation at the maximum thus gives a good indication of the match between the expected and actual pulse shapes and can be used to assess the quality of the TOF estimate. In practice (21.15) is used in discrete time form, where the integrals are replaced by sums of products and digital signal processors are an ideal implementation since they are highly optimized to perform this calculation [21.47, 48]. To achieve an arrival time estimator with resolution smaller than the discrete time sample rate, parabolic interpolation can be used on the maximum three correlations [21.11]. Of interest is the jitter standard deviation σ_R in the TOF estimator due to receiver noise. From [21.11, 46]

$$\sigma_R = \frac{\sigma_n}{B \sqrt{\sum_k \text{rec}(kT_s)^2}}, \quad (21.16)$$

where the summation index k is over the entire receiver pulse sampled every T_s seconds ($1 \mu\text{s}$ in [21.11, 47]), B is the bandwidth of the receiver pulse, and σ_n is the standard deviation of the receiver noise. Equation (21.16) shows that broadband high-energy pulses achieve low errors in the TOF estimator. In [21.11] this is achieved by using a 300 V pulse to excite the transmitter and achieve close to the impulse response from the device with a pulse shape similar to that shown in Fig. 21.6d.

21.10.2 Bearing Estimation

There are many proposed methods for bearing estimation. A single transducer [21.49] can be used by

exploiting the dependency of the received pulse shape on the angle of reception. This approach works for angles within one half of the beam width since the pulse shape is symmetric with respect to the transducer normal angle. Differences in zero-crossing times either side of the maximum amplitude of the pulse are used to obtain an accuracy of the order of 1° . Other single-receiver techniques rely on repeated measurements from a scan across the scene [21.50, 51] and achieve a similar level of accuracy but at much slower sensing speed since multiple readings are necessary.

Other single measurement approaches rely on two or more receivers [21.11, 12, 24]. This gives rise to a correspondence problem where data must be associated between the receivers. The closer the spacing between receivers, the simpler and more reliable the correspondence procedure. The misconception that bearing accuracy improves with larger receiver spacing ignores the correlation between measurement errors that can arise due to the measurements sharing an overlapping space of air in the propagation of the ultrasound. Due to the high accuracy of TOF estimation in [21.11] the receivers could be spaced as close as physically feasible (35 mm) and still bearing accuracies lower than any other systems are reported. Standard deviations of bearing errors are reported to be below 0.2° for a plane at a range of 4 m within a -10° to $+10^\circ$ beam width.

There are two common approaches to bearing estimation – interaural amplitude difference (IAD) [21.52] and interaural time difference (ITD) [21.11, 24, 47–49, 52]. IAD uses two receivers pointing away from each other so that an echo has a different amplitude response in each receiver's beam width. In ITD both receivers usually point in the same direction and the TOF is measured on each receiver and triangulation is applied to determine the angle of arrival. The bearing calculation is dependent on the target type, such as a plane, corner or edge; these geometries are analyzed in [21.11]. A simple arrangement with a transceiver and receiver is shown in Fig. 21.18, where T/R1 is the transceiver and R2 is the second receiver, spaced a distance d from each other. The virtual image of the transmitter is shown as T' . The two TOFs measured on the two receivers are t_1 and t_2 , and these are used to estimate the bearing angle, θ , to the plane which is the angle to the plane normal. Applying the cosine rule to the triangle $R2 R1 T'$ in Fig. 21.18 gives

$$\cos(90 - \theta) = \sin \theta = \frac{d^2 + c^2 t_1^2 - c^2 t_2^2}{2 d c t_1}. \quad (21.17)$$

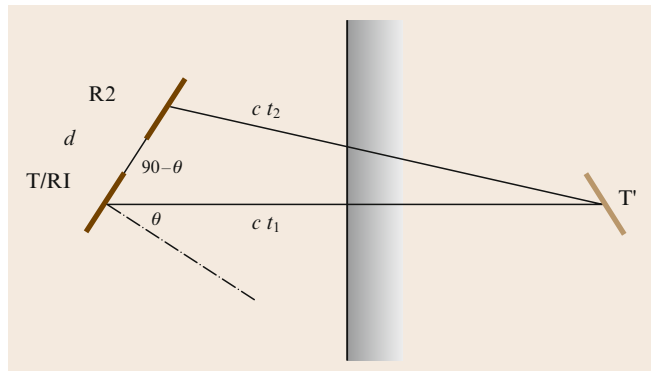


Fig. 21.18 Bearing (θ) calculation for a plane using a transceiver T/R1 and a receiver R2. T' is the virtual image of T

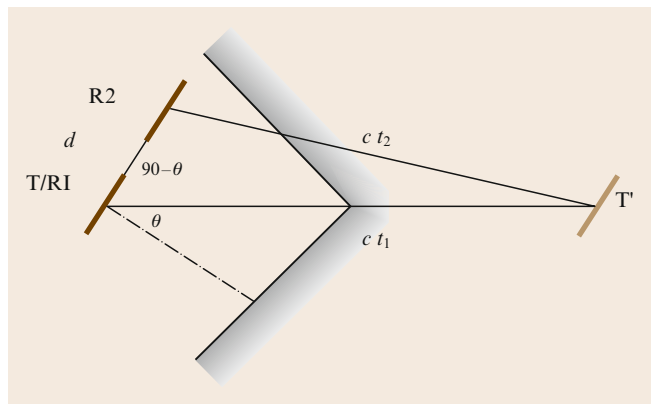


Fig. 21.19 Bearing (θ) calculation for a corner using a transceiver T/R1 and a receiver R2. T' is the virtual image of T

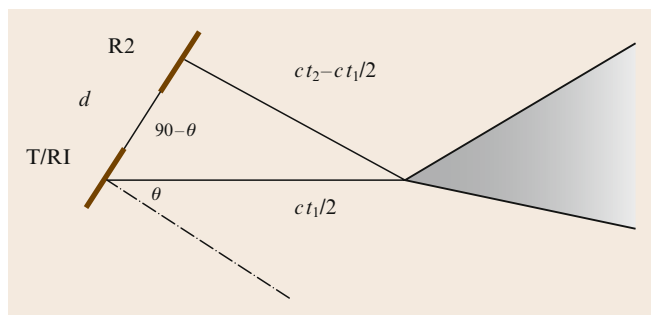


Fig. 21.20 Bearing (θ) calculation for an edge using a transceiver T/R1 and a receiver R2. No virtual image is present since the edge radiates from a point source at the edge

When $d \ll c t_1$, (21.17) can be approximated by

$$\sin \theta \approx \frac{c(t_1 - t_2)}{d}. \quad (21.18)$$

Note that any common (i. e., correlated) noise in t_1 and t_2 is removed by the difference in (21.18) and hence the correlation in noise components of the TOF cannot be overlooked in bearing estimation as described above.

The situation for a corner is shown in Fig. 21.19 and the same result applies as in (21.17). For an edge the situation is shown in Fig. 21.20, where R1 has a TOF from T to the edge and back to R1, whilst R2 has a TOF from T to the edge and back to R2. From the geometry,

we use the same approach as in (21.17) to give

$$\begin{aligned}\sin \theta &= \frac{d^2 + c^2 t_1^2 / 4 - c^2 (t_2 - t_1 / 2)^2}{2dct_1 / 2} \\ &= \frac{d^2 + c^2 t_2 (t_1 - t_2)}{dct_1} .\end{aligned}\quad (21.19)$$

Note that (21.19) can be approximated by (21.18) when $d \ll t_1$.

21.11 CTFM Sonar

Continuous-transmission frequency-modulated (CTFM) sonar differs from the more common pulse-echo sonar discussed in previous sections in the transmission coding and the processing required to extract information from the receiver signal.

21.11.1 CTFM Transmission Coding

The CTFM transmitter continuously emits a varying-frequency signal, usually based on a sawtooth pattern as shown in Fig. 21.21, where the frequency is often swept through an octave every sweep cycle T . The transmitted signal with a linearly changing frequency can be expressed as

$$S(t) = \cos[2\pi(f_H t - bt^2)] \quad (21.20)$$

for $0 \leq t < T$. The sweep cycle is repeated every T seconds as shown in Fig. 21.21. Frequency is $1/2\pi$ times the time derivative of the phase in (21.20). Note that the highest frequency is f_H and the lowest transmitted frequency is $f_H - 2bT$, where b is a constant that de-

termines the sweep rate. We can then define the swept frequency ΔF as

$$\Delta F = 2bT . \quad (21.21)$$

21.11.2 CTFM TOF Estimation

Echoes are generated when the transmitted wavefront encounters reflectors and are an attenuated, delayed version of the transmitted signal

$$E(t) = AS \left(t - \frac{2R}{c} \right) , \quad (21.22)$$

where R is the range to the reflector, c is the speed of sound and A is the amplitude that may in the case of curved objects depend on the frequency of the sound at reflection.

The TOF is estimated by the two step process of demodulation and spectral analysis. Demodulation is achieved by multiplying the received signal by a copy of the transmitted signal and low pass filtering. This can best be understood in the simple case of one echo. The signal $D(t)$ is obtained using (21.20) and (21.22)

$$\begin{aligned}D(t) &= E(t)S(t) \\ &= \frac{A}{2} [\cos(2\pi f_e t - \phi) \\ &\quad + \cos(2\pi f_u t - 2bt^2 - \phi)] \\ \text{for } f_e &= \frac{4Rb}{c} , \\ f_u &= \left(2f_H + \frac{4Rb}{c} \right) , \\ \phi &= f_H \frac{2R}{c} + \frac{4bR^2}{c^2} ,\end{aligned}\quad (21.23)$$

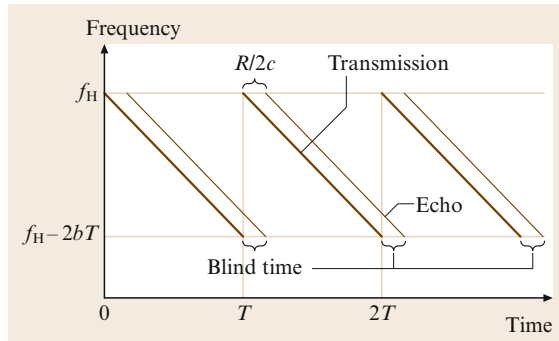


Fig. 21.21 Plot of frequency versus time for CTFM. The blind time applies if the shown echo corresponds to a maximum range target at R_m

where the following trigonometric identity has been used in (21.23)

$$\cos(x)\cos(y) = \frac{1}{2}[\cos(x-y) + \cos(x+y)] . \quad (21.24)$$

A low-pass filter removes frequency components above f_H and this results in the base-band signal:

$$D_b(t) = \frac{A}{2} \left[\cos \left(2\pi \frac{4Rb}{c} t - \phi \right) \right] \quad (21.25)$$

which has a frequency proportional to the range R . The ranges of echoes can be extracted by examining the spectrum of D_b using, for example, a discrete Fourier transform (DFT) or the fast Fourier transform (FFT) if the number of samples is a power of 2. From (21.25) for a frequency peak of f_r Hz the corresponding range R is given by

$$R = f_r \frac{c}{4b} . \quad (21.26)$$

Note that the above analysis relies on excluding the receiver waveform at the start of each sweep for a *blind time* (Fig. 21.21) of $R_m/2c$, where R_m is the maximum target range. During this blind time the receiver signal is dependent on the previous sweep rather than the current sweep as assumed in the analysis above. The sweep time T needs to be much larger than this blind time for the sonar to operate effectively. The blind time can be eliminated at the expense of introducing complexity in the demodulation process as described in [21.53], where an interlaced double demodulation scheme is described.

21.11.3 CTFM Range Discrimination and Resolution

We define range discrimination as the separation in range of two targets that can be simultaneously detected as distinct. The range resolution is defined as the smallest increment in range that can be measured by the sonar.

Suppose that, in order to extract ranges of targets, $D_b(t)$ from (21.25) is sampled at ΔT intervals and k samples are collected before a DFT (or FFT if k is a power of 2) is performed. The frequency samples of the DFT will be $\Delta f = 1/(k\Delta T)$ apart. From (21.26), this represents a range resolution ΔR of

$$\Delta R = \frac{c\Delta f}{4b} = \frac{c}{4bk\Delta T} . \quad (21.27)$$

We can relate this to the swept frequency ΔF from (21.21) as

$$\Delta R = \frac{c}{2\Delta F} \times \frac{T}{k\Delta T} , \quad (21.28)$$

where the second term is the ratio of the sweep time to the spectral sample time. In order to discriminate two peaks in the DFT, they must be at least two samples apart and hence

$$\text{range discrimination} = \frac{c}{\Delta F} \times \frac{T}{k\Delta T} . \quad (21.29)$$

Note that (21.28, 21.29) show that, subject to signal-to-noise constraints, CTFM can lengthen the *data integration time* $k\Delta T$ in order to improve the range discrimination and resolution of the sonar. Also it is possible, subject to signal noise, to use interpolation techniques (e.g., parabolic interpolation) on the DFT peaks to resolve to a smaller than Δf frequency and hence improve range resolution (but not range discrimination).

21.11.4 Comparison of CTFM and Pulse-Echo Sonar

- The range resolution of pulse-echo sonar and CTFM sonar is theoretically the same given the same signal-to-noise ratios and bandwidths [21.53]. The range discrimination in pulse-echo sonar is limited by the pulse length, where shorter pulse lengths require higher bandwidth. However in CTFM, range discrimination can be improved by increasing the data integration time, allowing more design flexibility.
- CTFM also allows for the energy of the transmitted signal to be spread evenly over time, resulting in lower peak acoustic power emission compared to pulse-echo systems with the same receiver signal-to-noise ratio. CTFM can provide a greater average power in a practical context and consequently a greater sensitivity to weak reflectors is possible.
- CTFM requires more-complex transmitter circuitry and the requirement for FFT processing on the receiver side.
- Separate transmitter and receiver transducers are necessary with CTFM, whilst pulse-echo systems can use a single transducer for both transmission and reception, resulting in restriction on the minimum range of pulse-echo sonar due to the blanking of the receiver during transmission. CTFM has no inherent restriction on minimum range.
- CTFM sonar can continuously derive range information from targets every $k\Delta T$ seconds at a delay of $R/c + k\Delta T$ compared to every $2R_m/c$ with a delay of $2R/c$ in pulse-echo sonar (ignoring processing

delays in both), which may be important in real-time tracking applications.

- Other benefits of CTFM are that the number of range measurements per cycle is limited only by the range discrimination constraint of (21.28) and the signal-to-noise ratio.
- In terms of bearing estimation and classification of targets from a moving platform, short pulse-echo sonar systems like [21.26, 47] do not suffer from the CTFM data integration time required to estimate accurately the frequencies corresponding to ranges (and hence bearing). During the data integration time, the target can move with respect to the sensor and blur the measurements, making bearing estimation and classification less accurate. In short pulse-echo systems, the target is effectively sampled with a pulse of less than 100 μs , resulting in a consistent snapshot of the target.

21.11.5 Applications of CTFM

Kay [21.54, 55] developed a mobility aid for blind people using a CTFM sonar system based on a sweep of $f_H = 100\text{ kHz}$ down to 50 kHz with a sweep period of $T = 102.4\text{ ms}$. After demodulation, ranges are heard as audible tones with frequencies up to 5 kHz corresponding to ranges of up to 1.75 m. The system uses one transmitter and three receivers as shown in Fig. 21.22. Users of the system can listen to the demodulated signal in stereo headphones corresponding to the left and right receivers, each mixed with the large central oval receiver. Higher frequencies correspond to more-distant ranges. To illustrate the sensitivity, a 1.5 mm-diameter wire is easily detectable at 1 m range – the echo produced is 35 dB above the noise floor in the system.

CTFM sonar has been used to recognize isolated plants [21.40, 56]. The advantage gained from CTFM is that extensive range and echo amplitude information is obtained from the whole plant given the spectrum of the demodulated received signal, and these echoes are obtained from an excitation across an octave of frequencies from 100 down to 50 kHz with a high signal-to-noise ratio that allows weak reflections from leaves to be sensed. This information is called the *acoustic density profile* and 19 different features are found to be useful in classifying the plants, such as the number of range cells above a threshold in amplitude, the sum of all range cells, the variation about the centroid, the distance from the first to the highest amplitude cell, and the range over which reflections are detected. With



Fig. 21.22 Aid for blind people – the small oval transducer is the transmitter and the other three components are receivers. The large oval receiver provides high resolution, enabling fixation by users' fine neck control (Photo courtesy [21.54])

a population of 100 plants, an average of 90.6% correct pairwise classification was obtained using a statistical classifier.

Scanning CTFM with a single transmitter and single receiver has been successfully applied to mapping of indoor environments that include smooth and rough surfaces [21.51] with bearing errors on the order of 0.5° for smooth surfaces and higher for edges. The classification uses amplitude information that is normalized with range using a fixed attenuation constant of sound. In practice this attenuation constant varies with temperature and humidity and needs to be calibrated before each experiment for consistent results. Greater robustness, speed, and accuracy has been demonstrated with TOF methods of classification that require at least two transmitter positions and two receivers as described in [21.11, 47]. CTFM could be applied to array systems to achieve higher sensitivity to weak targets than the existing pulse-echo systems.

CTFM has been employed in three binaural systems [21.12] where a rigorous theoretical and experimental comparison of these ultrasonic sensing systems based on different range and bearing estimators is made. Stanley [21.12] also contains detailed engineer-

ing design information of CTFM sonar systems. The conclusion is that CTFM can insonify large areas due to its higher average power transmissions and consequently good signal to noise performance. The use of autoregressive estimators for spectral lines in the demodulated signal were found to provide better res-

olution than the DFT. The interaural distance and power difference CTFM approaches provided state-of-the-art performance except that the pulse-echo approach in [21.11] using a high-energy short pulse was found to be a *factor of six to eight* times superior in bearing precision.

21.12 Multipulse Sonar

This section examines sonar systems that employ more than one pulse in the transmitter(s). The main motivations are interference rejection and on-the-fly classification. Multipulse sonar has also been used to generate a better signal-to-noise ratio by creating longer transmitted pulse sequence using Barker codes [21.57]. The autocorrelation of a Barker code gives a narrow peak with low autocorrelation away from the central lobe. The matched filter then gives rise to pulse compression that averages noise over a longer time period.

21.12.1 Interference Rejection

External acoustic noise, such as compressed air, is a source of sonar interference. Sonar systems attempt to reduce the effects of external interference by filtering the signal and the optimal filter is the matched filter where the impulse response is the time reversal of the pulse shape that is expected. Since a time-reversed convolution is a correlation, the matched filter then acts as a correlation with the expected pulse shape as discussed in Sect. 21.10. Approximations to matched filtering can be designed based on a bandpass filter with a frequency response that is similar to the spectrum of the expected receiver pulse. CTFM systems allow robust suppression of external interference by employing a matched filter across a broad range of frequencies contained in the continuous chirp transmission.

When more than one sonar system operates in the same environment, the transmitted signal from one sonar system can be received by another, causing crosstalk errors. This is particularly evident in classical sonar rings constructed from Polaroid ranging modules. Error-eliminating rapid ultrasonic firing strategies have been developed [21.58] and are claimed to remove most of this interference and allow faster operation of these sonar rings.

More-sophisticate coding of transmitted pulse(s) has been employed [21.22, 59–62] to allow rejection of external interference and crosstalk. One difficulty with

multiple transmitted pulses over an greater time period than a single pulse is that target clutter can produce many overlapping pulses at the receivers that are difficult to unravel and interpret, and the sonar range discrimination can be compromised.

21.12.2 On-the-Fly Target Classification

Target classification into planes, cylinders, and edges has been achieved by deploying a single transmitter and three receivers [21.24] using a single measurement cycle. At least two transmitters are required to differentiate planes from concave right-angled corners [21.11] where a two transmitter arrangement is used to classify targets into planes, corners, and edges in two successive measurement cycles. The method of classification can be understood with virtual images and mirrors, since specular sonar reflections occur. Looking into a plane mirror gives an image that is left–right reversed compared to looking into a right-angled mirror. An edge is analogous to observing a high-curvature specular surface, such as a polished chair leg, where the whole image is compressed to a point. Sonar classification exploits the difference in bearing angles to a target from two transmitters to classify as follows: a positive difference δ indicates a plane, a negative δ value indicates a corner, and zero difference an edge, where the angle δ depends on the sensor geometry and target range. More sophistication can be added by using range measurements in addition to bearing, with maximum-likelihood estimation.

This arrangement [21.11] was refined to work with just one measurement cycle of around 35 ms to 5 m range: hence the term *on-the-fly* in [21.47]. This on-the-fly approach uses pulses fired at a precise time difference ΔT and 40 mm apart from two transmitters with two further receivers completing a square. ΔT is usually around 200 μ s but can vary randomly from cycle to cycle to achieve interference rejection (both crosstalk and environmental) with identical sonar systems. Classification is performed simultaneously in one measurement cycle.

The sensor achieves high accuracy in range and bearing with robust classification by exploiting the tight correlation between TOF jitter in the different transmitter to

receiver paths due to the close temporal and spatial arrangement. The sensor has been deployed for large-scale mapping in [21.63].

21.13 Sonar Rings

21.13.1 Simple Ranging Module Rings

Since sonar only detects objects lying within its beam, a common means to scan the entire environment outside the robot is to use an array of sonars, or a *ring* [21.64]. The most common is the Denning ring that contains 24 sonars equally spaced around the robot periphery. This 15° spacing allows some overlap in the sonar beams so at least one of the sonars will detect a strong reflecting object. The sonars in the ring are typically employed sequentially one at a time. Using a 50 ms probing pulse period to reduce false readings a complete environmental scan is accomplished every 1.2 s. This sample time is adequate for a translate-and-stop operation in research settings, but may be too slow for a continually moving robot. A robot moving at 1 m/s may not detect an object with sufficient warning to prevent a collision. Some researchers propose simultaneously employing sonars on opposite ends of the

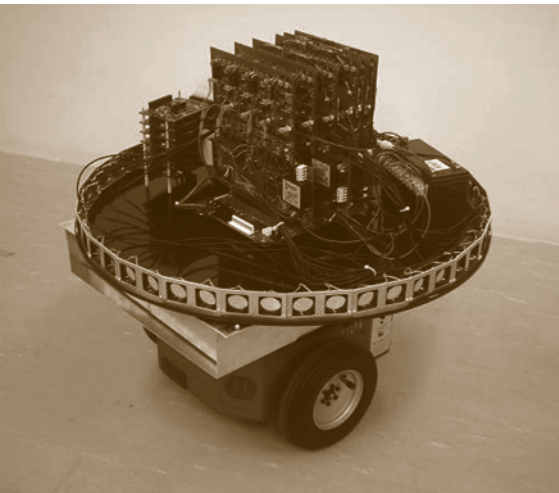
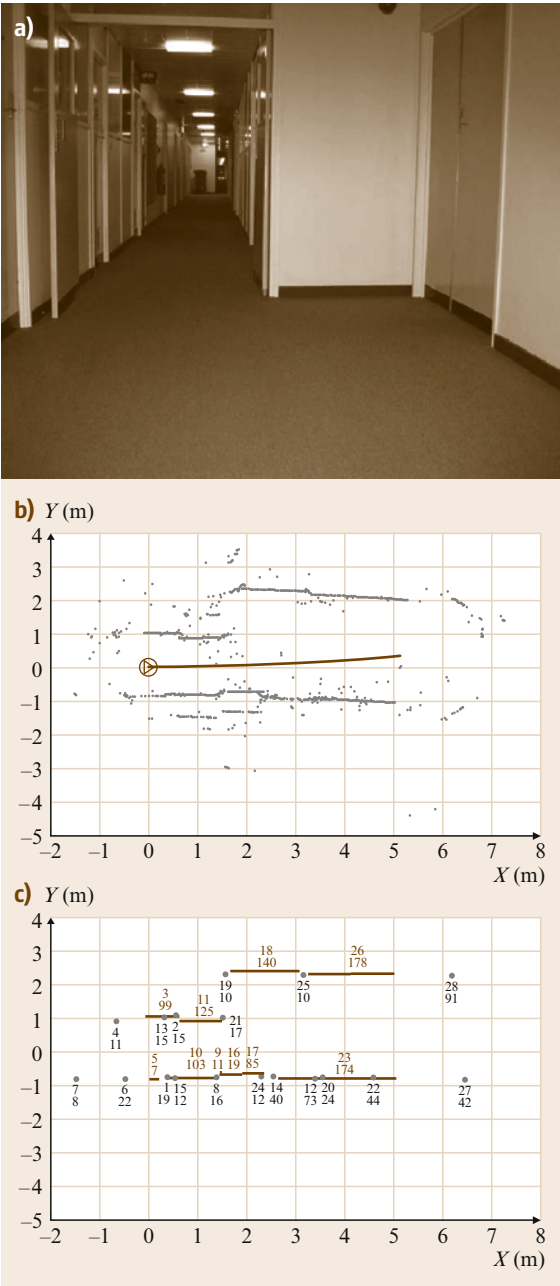


Fig. 21.23 DSP sonar ring hardware

Fig. 21.24a–c DSP sonar ring mapping an indoor environment (a), raw data (b) and SLAM feature map with feature number and number of associations shown as numbers (c). The sonar ring is moving at 10 cm/s with a 11.5 Hz sampling rate ►



ring to speed up acquisition times, while others also reduce the probing pulse period and attempt to identify artifacts.

21.13.2 Advanced Rings

Yata et al. [21.49] developed a 32 cm-diameter sonar ring with 30 transmitters and 30 receivers placed alternately. Murata piezoelectric MA40S4R wide-angle transducers are used to enable overlapping reception of echoes produced by firing all the transmitters simultaneously. An axial symmetrical exponential horn structure is used to narrow the beam shape of the transmitters vertically to avoid reflections from the floor. Received signals are compared with a decaying threshold to produce a 1 bit digitized sampled signal without rectification. Bearing is estimated from the leading edge of echoes and an error standard deviation of 0.4° is reported for ranges up to 1.5 m.

A sonar ring with seven digital signal processors (DSPs) [21.48, 65, 66] that uses 24 pairs of 7000 series Polaroid transducers consisting of a transceiver and receiver has been developed (Fig. 21.23). Each

pair can derive range and accurate bearing information using template-matching digital signal processing (Sect. 21.10) on each of the two receiver channels, which are sampled at 250 kHz with 12 bit analog to digital converters. In total eight receiver channels are processed per DSP. All transceivers are fired simultaneous to enable full surrounding sensing of the environment approximately 11 times a second to a 6 m range with experimentally validated range and bearing accuracies to smooth targets of 0.6 mm and 0.2° , respectively. To suppress interference between neighboring pairs, two different transmitted pulse shapes are employed in an interleaved fashion around the perimeter of the ring. The pulse shapes are derived from two and three cycles of 65 kHz excitation. The DSP sonar ring allows rapid and accurate wall following, map building, and obstacle avoidance due to the high repetition and accurate range and bearing sensing. The beam width of the transducer pairs allows full 360° coverage with respect to smooth specular targets to a range of 3 m. An example of the DSP sonar ring producing a feature for a simultaneous localization and mapping (SLAM) map is shown in Fig. 21.24.

21.14 Motion Effects

When a sensor moves with respect to its targets, sonar measurements are effected. For example, a sonar sensor moving at a speed of 1% of the speed of sound (around 3.4 m/s) will experience errors of the order of 0.6° for some bearing measurements. The effects of linear velocity on the TOF and reception angle are dependent on the target type and hence for motion compensation to be meaningful a target classification sensor is needed. We consider the classical plane, edge, and corner target types in this section. Rotational motion effects are discussed in [21.26] where it is shown that very high speeds of rotation are necessary to give rise to a small bearing error (e.g., 0.1° error for approximately 1700 deg/s). Narrowing of the effective beam width is another effect of high rotation speeds of a sonar sensor.

The sensor is assumed to transmit from a point labeled T and receiver measurements are referenced to this position on the sensor. However, due to the motion of the sensor, the ground referenced position R at the time of reception of the echo moves from T over the course of the TOF. For a linear velocity, the distance between T and R is $\text{TOF} \times v$, where v is the magnitude of the sensor velocity vector relative to the ground, with com-

ponents v_x and v_y parallel to their respective coordinate axes. The expressions derived for linear motion apply to any sonar sensor, since only the physics of sound propagation and reflection are used. All targets are assumed to be stationary.

This section is based on [21.26], where further experimental work not included here can be found.

21.14.1 Moving Observation of a Plane

A plane target reflects the transmission from position T to R as shown in Fig. 21.25a. The TOF is broken up into two parts: t_1 is the time of propagation to the plane and t_2 from the plane to the receiver R. Here we derive the effect of linear motion on the TOF = $t_1 + t_2$ and the angle of reception θ all taken from the view of a stationary observer. A moving observer is discussed below.

From the right-angle triangle on the left of Fig. 21.25a, we have

$$\sin \theta = \frac{v_x}{c} \text{ and } \cos \theta = \sqrt{1 - \left(\frac{v_x}{c}\right)^2} \quad (21.30)$$

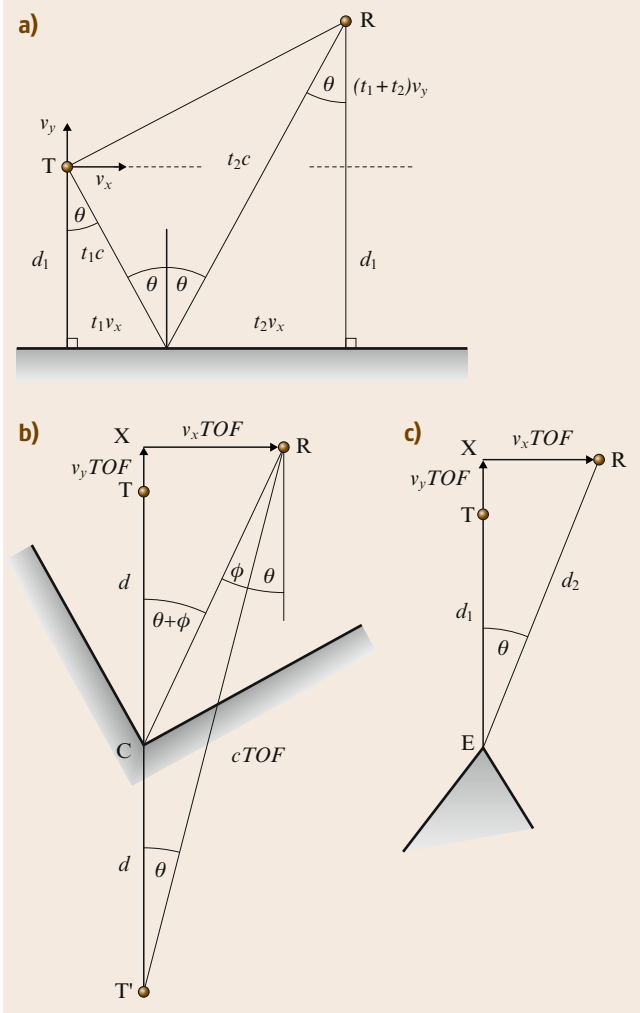


Fig. 21.25a–c Observing a target from a moving sensor. T is the position of the transmitter and R is where the echo is received at the end of the TOF. The target is a plane in (a), a corner in (b), and an edge in (c)

and also

$$\cos \theta = \frac{d_1}{t_1 c} \implies t_1 = \frac{d_1}{c \cos \theta} . \quad (21.31)$$

From the right-angled triangle on the right of Fig. 21.25a, we have

$$\cos \theta = \frac{(t_1 + t_2)v_y + d_1}{t_2 c} \implies t_2 = \frac{(t_1 + t_2)v_y + d_1}{c \cos \theta} . \quad (21.32)$$

The TOF is obtained by adding (21.31) and (21.32) and then substituting (21.31) giving

$$\text{TOF} = \left(\frac{2d_1}{c} \right) \frac{1}{\sqrt{1 - \frac{v_x^2}{c^2} - \frac{v_y}{c}}} . \quad (21.33)$$

The first factor in (21.33) represents the stationary TOF. The second factor approaches unity as the velocity approaches zero.

21.14.2 Moving Observation of a Corner

Figure 21.25b shows the situation for a corner with the virtual image of T is shown as T'. From the right-angled triangle T'XR

$$c^2 \text{TOF}^2 = (2d_1 + v_y \text{TOF})^2 + v_x^2 \text{TOF}^2 \quad (21.34)$$

which gives

$$\text{TOF} = \frac{2d_1}{c} \left[\frac{\sqrt{1 - \left(\frac{v_x}{c}\right)^2} + \frac{v_y}{c}}{1 - \left(\frac{v_y}{c}\right)^2} \right] , \quad (21.35)$$

where $v^2 = v_x^2 + v_y^2$. The left-hand term of (21.35) is the stationary TOF and the right-hand term approaches unity for small velocities. The angle ϕ in Fig. 21.25b is the angle deviation due to motion as reference by a stationary observer. From the triangles T'XR and CXR

$$\tan \theta = \frac{v_x \text{TOF}}{2d_1 + v_y \text{TOF}} \quad \text{and}$$

$$\tan(\theta + \phi) = \frac{v_x \text{TOF}}{d_1 + v_y \text{TOF}} . \quad (21.36)$$

From (21.36), we have

$$\tan(\theta + \phi) = \left(2 - \frac{v_y \text{TOF}}{d_1 + v_y \text{TOF}} \right) \tan \theta \quad (21.37)$$

and solving for $\tan \phi$ yields

$$\begin{aligned} \tan \phi &= \tan \theta \left(\frac{1 - \sin^2 \theta}{\frac{v_y \text{TOF}}{d_1} + 1 + \sin^2 \theta} \right) \\ &= \left(\frac{v_x}{\frac{2d_1}{\text{TOF}} + v_y} \right) \left(\frac{1 - \sin^2 \theta}{\frac{v_y \text{TOF}}{d_1} + 1 + \sin^2 \theta} \right) . \end{aligned} \quad (21.38)$$

For $v_x, v_y \ll c$, $\sin \theta \ll 1$ and $2d_1/\text{TOF} \approx c$ we can approximate (21.38) as

$$\phi \approx \frac{v_x}{c} . \quad (21.39)$$

21.14.3 Moving Observation of a Edge

Since an edge reradiates the incoming ultrasound from an effective point source, the reception angle with respect to a stationary observer is unaffected by motion as shown in Fig. 21.25c. The TOF is affected due to the motion moving the receiving position. From the right-angled triangle XER, $d_2^2 = (d_1 + v_y)^2 + v_x^2 \text{TOF}^2$ and $d_1 + d_2 = c \text{TOF}$ leads to

$$\text{TOF} = \frac{2d_1}{c} \left(\frac{1 + \frac{v_y}{c}}{1 - \frac{v^2}{c^2}} \right) \approx \frac{2d_1}{c} \left(1 + \frac{v_y}{c} \right), \quad (21.40)$$

where the approximate holds in (21.40) for $v \ll c$.

21.14.4 The Effect of a Moving Observation on the Angle of Reception

The expressions for the reception angle in the previous sections are based on an observer that is stationary with respect to the propagating medium air. In practice the observer is the sensor, and is moving with a velocity v . Suppose that the sonar wave arrives at an angle α relative to the air, as shown in Fig. 21.26. The velocity components of the wavefront relative to the observer, w_x and w_y , are as follows

$$w_x = c \sin \alpha - v_x \text{ and } w_y = c \cos \alpha - v_y. \quad (21.41)$$

From (21.41) the observed angle of arrival, β is

$$\tan \beta = \frac{c \sin \alpha - v_x}{c \cos \alpha - v_y} = \frac{\sin \alpha - \frac{v_x}{c}}{\cos \alpha - \frac{v_y}{c}}. \quad (21.42)$$

21.14.5 Plane, Corner, and Edge Moving Observation Arrival Angles

In this section the arrival angles (in radians) for each target type are summarized and approximated for speeds expected of a mobile robot. The speed is assumed

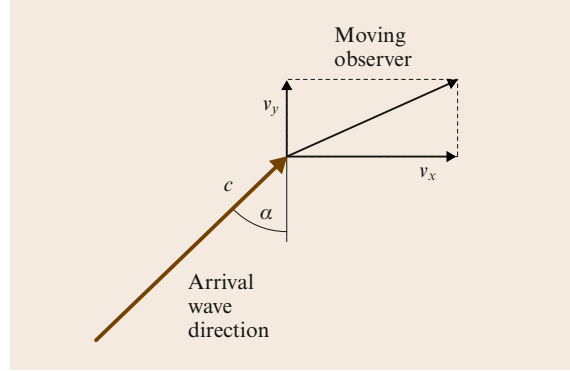


Fig. 21.26 Observation of arriving wave from a moving observer

to be less than a few percent of the speed of sound (typically 340 m/s at room temperature). These effects have been observed experimentally at speeds of up to 1 m/s [21.26].

Equations (21.41) and (21.30) cancel exactly, and for a plane the arrival angle relative to the sensor is exactly zero

$$\beta_{\text{plane}} = 0. \quad (21.43)$$

This can be explained by noting that the forward wave velocity component is always the same as the sensor's as reflection preserves this component.

For a corner the angle ϕ results in a wavefront that appears to be displaced in the same direction as the sensor motion from the real corner direction, as can be seen in Fig. 21.25b. The effect of the moving observer doubles this effect as seen by (21.41) and (21.39)

$$\beta_{\text{corner}} \approx -\frac{2v_x}{c}. \quad (21.44)$$

For an edge the result is due to the observer only

$$\beta_{\text{edge}} \approx \tan^{-1} \left(\frac{0 - \frac{v_x}{c}}{\cos \alpha - \frac{v_y}{c}} \right) \approx -\frac{v_x}{c}. \quad (21.45)$$

21.15 Biomimetic Sonars

The success of biosonars, bats and dolphins [21.67], has led researchers to implement sonars based on biosonar morphology, strategy, and nonlinear processing. The capabilities exhibited by biosonars have caused researchers to examine biomimicking (*biomimetic*) systems.

Biosonar morphology typically has a single transmitter and a pair of receivers. Bats transmit sound pulses through the mouth or nose, while dolphins transmit through a melon. The two receivers correspond to ears that permit binaural processing. Mimicking binaural hearing has led to small arrays that localize objects [21.8]



Fig. 21.27 Biomimetic configuration sonar with center transmitter flanked by receivers that rotate

and scanning strategies [21.68]. Movable pinnae observed in bats have motivated research in receivers that rotate [21.69, 70]. Figure 21.27 shows one such example.

Rotating the receivers so their axes fall onto the reflecting object not only increases the detected echo amplitude, but also its bandwidth, both effects improving the ability to classify an object.

Biosonar strategy provides clues for successful object localization. It is well known that the object location within the transducer beam affects the echo waveform and complicates the inverse problem of object classification [21.10, 71]. Dolphin movies show that they maneuver to position an object at a repeatable location and range, guided by binaural echo processing. This has motivated a dolphin-mimicking movable sonar positioned at the end of a robot arm for object classification [21.10, 71], as shown in Fig. 21.28. This system was able to differentiate reliably the head and tail side of a coin, but only after introducing a scan in elevation to accommodate for the lack of such positioning afforded by binaural hearing. The idea for a scan over elevation was motivated by the nodding motion that dolphins exhibit when searching for prey lying under the sand.

Another useful strategy, suggested by probing pulses emitted by bats, is processing echo sequences. As an

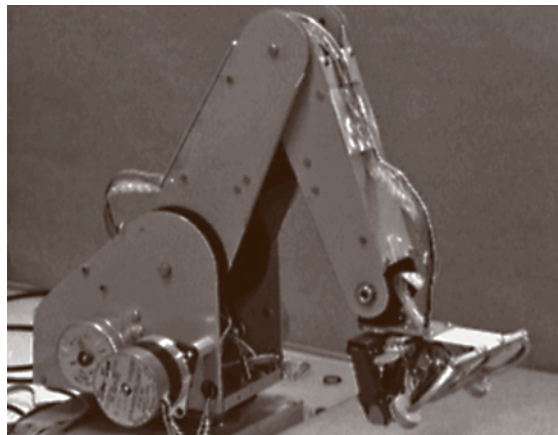


Fig. 21.28 Biomimetic sonar mounted on the end of a robot arm

extension to the conventional stop-and-scan operation of most sonars, sonar data were acquired while the sonar was moving along piecewise linear paths to reveal hyperbolic trends, similar to acoustic flow [21.72]. Matching data to hyperbolic trends permits the estimation of the passing range, which is useful for collision avoidance and passing through narrow openings [21.72].

Most sonar systems use classical estimation procedures involving correlation detection and spectrum analysis. The cochlear model has led to multiple band-pass filters to process wide-band pulses for environmental landmark classification [21.73]. The action potential spikes observed in the biological nervous system also suggest neuromorphic processing based on coincidence detection. The sparse information provided by conventional TOF measurements motivated sonar detectors that provided complete echo waveform information from multiple detections that result in spike-like data [21.23, 74]. Applying temporal and spatial coincidence to spike data has led to reverberation artifact recognition [21.35] and passing-range estimation [21.75].

Such biomimetic techniques provide insights into the information content present in echoes and the type of sensing tasks for which sonar is best suited.

21.16 Conclusions

Sonar is a useful, inexpensive, low-power, light-weight and simple ranging sensor for robotics applications

that can provide accurate object localisation. For sonar to be effectively employed, understanding its physical

principles and implementation is important and these topics are covered in the early sections of this chapter. Various approaches to sonar sensing are highlighted from simple single-transducer ranging to more sophisticated multi-transducer and multi-pulse configurations with associated signal processing requirements. Sophisticated sonars are capable of measuring target range and angle accurately as well as classifying targets, reject-

ing interference, and compensating for motion. Sonar rings provide surrounding environmental coverage, and CTFM systems improve the sensitivity for detecting small reflectors. Research is ongoing in areas such as signal and data processing, sonar map building, sonar configurations, transducer technology and biomimetic sonar that draw inspiration from biological sonar systems, such as used by bats and dolphins.

References

- 21.1 L.E. Kinsler, A.R. Frey, A.B. Coppens, J.V. Sanders: *Fundamentals of Acoustics* (Wiley, New York 1982)
- 21.2 R.C. Weast, M.J. Astle (Eds.): *CRC Handbook of Chemistry and Physics*, 59th edn. (CRC, Boca Raton 1978)
- 21.3 J. Borenstein, H.R. Everett, L. Feng: *Navigating Mobile Robots* (Peters, Wellesley 1996)
- 21.4 R. Kuc, M.W. Siegel: Physically-based simulation model for acoustic sensor robot navigation, *IEEE Trans. Pattern Anal. Mach. Intell.* **9**(6), 766–778 (1987)
- 21.5 SensComp: *7000 Series* (SensComp, Livonia 2007), <http://www.senscomp.com>
- 21.6 H.H. Poole: *Fundamentals of Robotics Engineering* (Van Nostrand, New York 1989)
- 21.7 J.E. Piercy: *American National Standard: Method for Calculation of the Absorption of Sound by the Atmosphere ANSI SI-26-1978* (Acoust. Soc. Am., Washington 1978)
- 21.8 B. Barshan, R. Kuc: A bat-like sonar system for obstacle localization, *IEEE Trans. Syst. Man Cybern.* **22**(4), 636–646 (1992)
- 21.9 R. Kuc: Three dimensional docking using qualitative sonar. In: *Intelligent Autonomous Systems IAS-3*, ed. by F.C.A. Groen, S. Hirose, C.E. Thorpe (IOS, Washington 1993) pp. 480–488
- 21.10 R. Kuc: Biomimetic sonar locates and recognizes objects, *J. Ocean. Eng.* **22**(4), 616–624 (1997)
- 21.11 L. Kleeman, R. Kuc: Mobile robot sonar for target localization and classification, *Int. J. Robot. Res.* **14**(4), 295–318 (1995)
- 21.12 B. Stanley: A comparison of binaural ultrasonic sensing systems. Ph.D. Thesis (University of Wollongong, Wollongong 2003), <http://adt.caul.edu.au/>
- 21.13 F.L. Degertekin, S. Calmes, B.T. Khuri-Yakub, X. Jin, I. Ladabaum: Fabrication and characterization of surface micromachined capacitive ultrasonic immersion transducers, *J. Microelectromech. Syst.* **8**(1), 100–114 (1999)
- 21.14 B. Barshan, R. Kuc: Differentiating sonar reflections from corners and planes by employing an intelligent sensor, *IEEE Trans. Pattern Anal. Mach. Intell.* **12**(6), 560–569 (1990)
- 21.15 A. Freedman: A mechanism of acoustic echo formation, *Acustica* **12**, 10–21 (1962)
- 21.16 A. Freedman: The high frequency echo structure of some simple body shapes, *Acustica* **12**, 61–70 (1962)
- 21.17 Ö. Bozma, R. Kuc: A physical model-based analysis of heterogeneous environments using sonar – ENDURA method, *IEEE Trans. Pattern Anal. Mach. Intell.* **16**(5), 497–506 (1994)
- 21.18 Ö. Bozma, R. Kuc: Characterizing pulses reflected from rough surfaces using ultrasound, *J. Acoust. Soc. Am.* **89**(6), 2519–2531 (1991)
- 21.19 P.J. McKerrow: Echolocation – from range to outline segments. In: *Intelligent Autonomous Systems IAS-3*, ed. by F.C.A. Groen, S. Hirose, C.E. Thorpe (IOS, Washington 1993) pp. 238–247
- 21.20 J. Thomas, C. Moss, M. Vater (Eds.): *Echolocation in Bats and Dolphins* (University of Chicago Press, Chicago 2004)
- 21.21 J. Borenstein, Y. Koren: Error eliminating rapid ultrasonic firing for mobile robot obstacle avoidance, *IEEE Trans. Robot. Autom.* **11**(1), 132–138 (1995)
- 21.22 L. Kleeman: Fast and accurate sonar trackers using double pulse coding, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.* (1999) pp. 1185–1190
- 21.23 R. Kuc: Pseudo-amplitude sonar maps, *IEEE Trans. Robot. Autom.* **17**(5), 767–770 (2001)
- 21.24 H. Peremans, K. Audenaert, J.M. Van Campenhout: A high-resolution sensor based on tri-aural perception, *IEEE Trans. Robot. Autom.* **9**(1), 36–48 (1993)
- 21.25 A. Sabatini, O. Di Benedetto: Towards a robust methodology for mobile robot localization using sonar, *IEEE Int. Conf. Robot. Autom.* (1994) pp. 3136–3141
- 21.26 L. Kleeman: Advanced sonar with velocity compensation, *Int. J. Robot. Res.* **23**(2), 111–126 (2004)
- 21.27 A. Elfes: Sonar-based real world mapping and navigation, *IEEE Trans. Robot. Autom.* **3**, 249–265 (1987)
- 21.28 S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg,

- N. Roy, J. Schulte, D. Schulz: MINERVA: A second generation mobile tour-guide robot, IEEE Int. Conf. Robot. Autom. (1999) pp. 3136–3141
- 21.29 K. Konolige: Improved occupancy grids for map building, Auton. Robot. **4**, 351–367 (1997)
- 21.30 R. Grabowski, P. Khosla, H. Choset: An enhanced occupancy map for exploration via pose separation, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003) pp. 705–710
- 21.31 J.D. Tardos, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, Int. J. Robot. Res. **21**(6), 311–330 (2002)
- 21.32 O. Aycard, P. Larouche, F. Charpillet: Mobile robot localization in dynamic environments using places recognition, Proc. IEEE Int. Conf. Robot. Autom. (1998) pp. 3135–3140
- 21.33 B. Kuipers, P. Beeson: Bootstrap learning for place recognition, Proc. 18-th Nat. Conf. Artif. Intell. (AAAI-02) (2002)
- 21.34 A. Bandera, C. Urdiales, F. Sandoval: Autonomous global localization using Markov chains and optimized sonar landmarks, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2000) pp. 288–293
- 21.35 R. Kuc: Biomimetic sonar and neuromorphic processing eliminate reverberation artifacts, IEEE Sens. J. **7**(3), 361–369 (2007)
- 21.36 A.M. Sabatini: A stochastic model of the time-of-flight noise in airborne sonar ranging systems, IEEE Trans. Ultrason. Ferroelectr. Freq. Control **44**(3), 606–614 (1997)
- 21.37 C. Biber, S. Ellin, E. Sheck, J. Stempeck: The Polaroid ultrasonic ranging system, Proc. 67th Audio Eng. Soc. Convention (1990)
- 21.38 R. Kuc: Forward model for sonar maps produced with the Polaroid ranging module, IEEE Trans. Robot. Autom. **19**(2), 358–362 (2003)
- 21.39 M.K. Brown: Feature extraction techniques for recognizing solid objects with an ultrasonic range sensor, IEEE J. Robot. Autom. **RA-1**(4), 191–205 (1985)
- 21.40 N.L. Harper, P.J. McKerrow: Classification of plant species from CTFM ultrasonic range data using a neural network, Proc. IEEE Int. Conf. Neural Netw. (1995) pp. 2348–2352
- 21.41 Z. Politis, P.J. Probert: Target localization and identification using CTFM sonar imaging: The AURBIT method, Proc. IEEE Int. Symp. CIRA (1999) pp. 256–261
- 21.42 R. Mueller, R. Kuc: Foliage echoes: A probe into the ecological acoustics of bat echolocation, J. Acoust. Soc. Am. **108**(2), 836–845 (2000)
- 21.43 P.N.T. Wells: *Biomedical Ultrasonics* (Academic, New York 1977)
- 21.44 J.L. Prince, J.M. Links: *Medical Imaging Signals and Systems* (Pearson Prentice Hall, Upper Saddle River 2006)
- 21.45 J.J. Leonard, H.F. Durrant-Whyte: Mobile robot localization by tracking geometric beacons, IEEE Trans. Robot. Autom. **7**(3), 376–382 (1991)
- 21.46 P.M. Woodward: *Probability and Information Theory with Applications to Radar*, 2nd edn. (Pergamon, Oxford 1964)
- 21.47 A. Heale, L. Kleeman: Fast target classification using sonar, IEEE/RSJ Int. Conf. Robot. Syst. (2001) pp. 1446–1451
- 21.48 S. Fazli, L. Kleeman: A real time advanced sonar ring with simultaneous firing, Proc. IEEE/RSJ Intern. Conf. Intell. Robot. Syst. (2004) pp. 1872–1877
- 21.49 T. Yata, A. Ohya, S. Yuta: A fast and accurate sonar-ringing sensor for a mobile robot, Proc. IEEE Int. Conf. Robot. Autom. (1999) pp. 630–636
- 21.50 L. Kleeman: Scanned monocular sonar and the doorway problem, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (1996) pp. 96–103
- 21.51 G. Kao, P. Probert: Feature extraction from a broadband sonar sensor for mapping structured environments efficiently, Int. J. Robot. Res. **19**(10), 895–913 (2000)
- 21.52 B. Stanley, P. McKerrow: Measuring range and bearing with a binaural ultrasonic sensor, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (1997) pp. 565–571
- 21.53 P.T. Gough, A. de Roos, M.J. Cusdin: Continuous transmission FM sonar with one octave bandwidth and no blind time. In: *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer-Verlag, Berlin, Heidelberg 1990) pp. 117–122
- 21.54 L. Kay: A CTFM acoustic spatial sensing technology: its use by blind persons and robots, Sens. Rev. **19**(3), 195–201 (1999)
- 21.55 L. Kay: Auditory perception and its relation to ultrasonic blind guidance aids, J. Br. Inst. Radio Eng. **24**, 309–319 (1962)
- 21.56 P.J. McKerrow, N.L. Harper: Recognizing leafy plants with in-air sonar, IEEE Sens. **1**(4), 245–255 (2001)
- 21.57 K. Audenaert, H. Peremans, Y. Kawahara, J. Van Campenhout: Accurate ranging of multiple objects using ultrasonic sensors, Proc. IEEE Int. Conf. Robot. Autom. (1992) pp. 1733–1738
- 21.58 J. Borenstein, Y. Koren: Noise rejection for ultrasonic sensors in mobile robot applications, Proc. IEEE Int. Conf. Robot. Autom. (1992) pp. 1727–1732
- 21.59 K.W. Jorg, M. Berg: Mobile robot sonar sensing with pseudo-random codes, Proc. IEEE Int. Conf. Robot. Autom. (1998) pp. 2807–2812
- 21.60 S. Shoval, J. Borenstein: Using coded signals to benefit from ultrasonic sensor crosstalk in mobile robot obstacle avoidance, Proc. IEEE Int. Conf. Robot. Autom. (2001) pp. 2879–2884
- 21.61 K. Nakahira, T. Kodama, T. Furuhashi, H. Maeda: Design of digital polarity correlators in a multiple-

- user sonar ranging system, IEEE Trans. Instrum. Meas. **54**(1), 305–310 (2005)
- 21.62 A. Heale, L. Kleeman: A sonar sensor with random double pulse coding, Aust. Conf. Robot. Autom. (2000) pp. 81–86
- 21.63 A. Diosi, G. Taylor, L. Kleeman: Interactive SLAM using Laser and Advanced Sonar, Proc. IEEE Int. Conf. Robot. Autom. (2005) pp. 1115–1120
- 21.64 S.A. Walter: The sonar ring: obstacle detection for a mobile robot, Proc. IEEE Int. Conf. Robot. Autom. (1987) pp. 1574–1578
- 21.65 S. Fazli, L. Kleeman: Wall following and obstacle avoidance results from a multi-DSP sonar ring on a mobile robot, Proc. IEEE Int. Conf. Mechatron. Autom. (2005) pp. 432–436
- 21.66 S. Fazli, L. Kleeman: Sensor design and signal processing for an advanced sonar ring, Robotica **24**(4), 433–446 (2006)
- 21.67 W.W.L. Au: *The Sonar of Dolphins* (Springer-Verlag, Berlin, Heidelberg 1993)
- 21.68 B. Barshan, R. Kuc: Bat-like sonar system strategies for mobile robots, Proc. IEEE Int. Conf. Syst. Man Cybern. (1991)
- 21.69 R. Kuc: Biologically motivated adaptive sonar, J. Acoust. Soc. Am. **100**(3), 1849–1854 (1996)
- 21.70 V.A. Walker, H. Peremans, J.C.T. Hallam: One tone, two ears, three dimensions: A robotic investigation of pinnae movements used by rhinolophid and hipposiderid bats, J. Acoust. Soc. Am. **104**, 569–579 (1998)
- 21.71 R. Kuc: Biomimetic sonar system recognizes objects using binaural information, J. Acoust. Soc. Am. **102**(2), 689–696 (1997)
- 21.72 R. Kuc: Recognizing retro-reflectors with an obliquely-oriented multi-point sonar and acoustic flow, Int. J. Robot. Res. **22**(2), 129–145 (2003)
- 21.73 R. Mueller, R. Kuc: Foliage echoes: A probe into the ecological acoustics of bat echolocation, J. Acoust. Soc. Am. **108**(2), 836–845 (2000)
- 21.74 T. Horiuchi, T. Swindell, D. Sander, P. Abshire: A low-power CMOS neural amplifier with amplitude measurements for spike sorting, ISCAS, Vol. IV (2004) pp. 29–32
- 21.75 R. Kuc: Neuromorphic processing of moving sonar data for estimating passing range, IEEE Sens. J. – Special Issue on Intelligent Sensors **7**(5), 851–859 (2007)

Range Sensors

Robert B. Fisher, Kurt Konolige

Range sensors are devices that capture the three-dimensional (3-D) structure of the world from the viewpoint of the sensor, usually measuring the depth to the nearest surfaces. These measurements could be at a single point, across a scanning plane, or a full image with depth measurements at every point. The benefits of this range data is that a robot can be reasonably certain where the real world is, relative to the sensor, thus allowing the robot to more reliably find navigable routes, avoid obstacles, grasp objects, act on industrial parts, etc.

This chapter introduces the main representations for range data (point sets, triangulated surfaces, voxels), the main methods for extracting usable features from the range data (planes, lines, triangulated surfaces), the main sensors for acquiring it (Sect. 22.1 – stereo and laser triangulation and ranging systems), how multiple observations of the scene, e.g., as if from a moving robot, can be registered (Sect. 22.2), and several indoor and outdoor robot applications where range data greatly simplifies the task (Sect. 22.3).

22.1 Range Sensing Basics	521
22.1.1 Range Images and Point Sets	521
22.1.2 Stereo Vision	523
22.1.3 Laser-Based Range Sensors	527
22.1.4 Time-of-Flight Range Sensors	528
22.1.5 Modulation Range Sensors	529
22.1.6 Triangulation Range Sensors	529
22.1.7 Example Sensors	530
22.2 Registration	530
22.2.1 3-D Feature Representations	530
22.2.2 3-D Feature Extraction	532
22.2.3 Model Matching and Multiple-View Registration	533
22.2.4 Maximum-Likelihood Registration	534
22.2.5 Multiple-Scan Registration	535
22.2.6 Relative Pose Estimation	535
22.2.7 3-D Applications	536
22.3 Navigation and Terrain Classification	537
22.3.1 Indoor Reconstruction	537
22.3.2 Urban Navigation	537
22.3.3 Rough Terrain	539
22.4 Conclusions and Further Reading	540
References	540

22.1 Range Sensing Basics

Here we present: (1) the basic representations used for range image data, (2) a brief introduction to the main 3-D sensors that are less commonly used in robotics applications, and (3) a detailed presentation of the commoner laser-based range image sensors.

22.1.1 Range Images and Point Sets

Range data is a $2\frac{1}{2}$ -D or 3-D representation of the scene around the robot. The 3-D aspect arises because we are measuring the (X, Y, Z) coordinates of one or more points in the scene. Often only a sin-

gle range image is used at each time instance. This means that we only observe the front sides of objects – the portion of the scene visible from the robot. In other words, we do not have a full 3-D observation of all sides of a scene. This is the origin of the term $2\frac{1}{2}$ -D. Figure 22.1a shows a sample range image and Fig. 22.1b shows a registered reflectance image, where each pixel records the level of reflected infrared light.

There are two standard formats for representing range data. The first is an image $d(i, j)$, which records the distance d to the corresponding scene point (X, Y, Z)

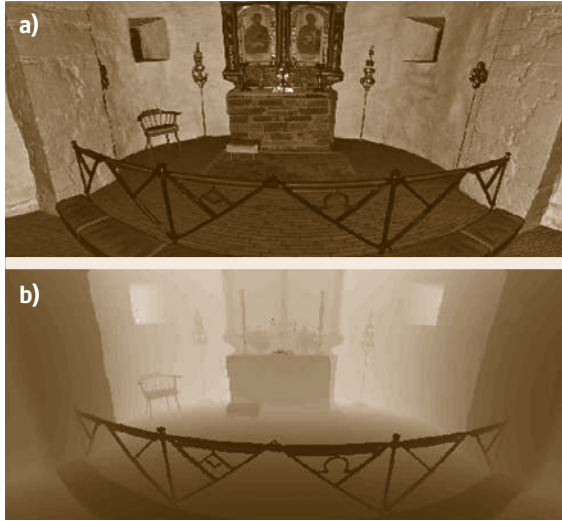


Fig. 22.1 (a) Registered infrared reflectance image. (b) Range image where closer is darker

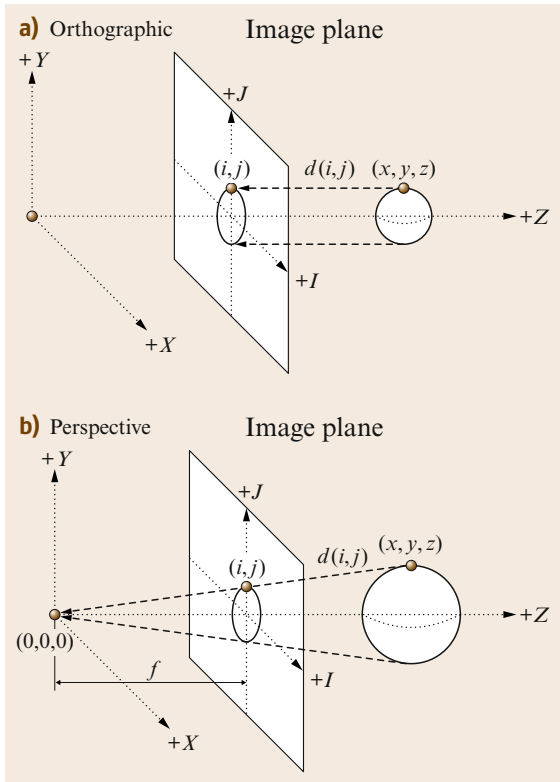


Fig. 22.2a,b Different range image mappings: (a) orthographic and (b) perspective

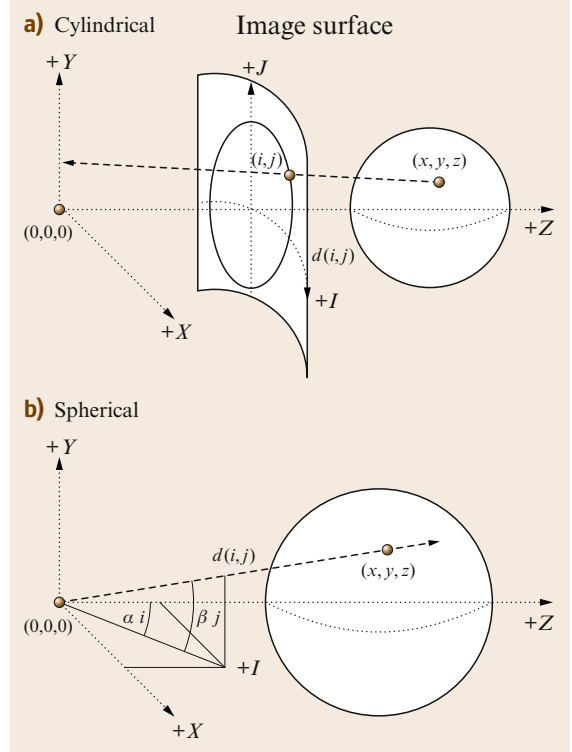


Fig. 22.3a,b Different range image mappings: (a) cylindrical and (b) spherical

for each image pixel (i, j) . There are several common mappings from $(i, j, d(i, j))$ to (X, Y, Z) , usually arising from the geometry of the range sensor. The most common image mappings are illustrated in Figs. 22.2 and 22.3. In the formulas given here, α and β are calibrated values specific to the sensor.

1. *Orthographic*. Here $(X, Y, Z) = (\alpha i, \beta j, d(i, j))$. These images often arise from range sensors that scan by translating in the x and y directions. (Fig. 22.2a.)
2. *Perspective*. Here $d(i, j)$ is the distance along the line of sight of the ray through pixel (i, j) to point (x, y, z) . Treating the range sensor focus as the origin $(0, 0, 0)$ and assuming that its optical axis is the Z axis and that the (X, Y) axes are parallel to the image (i, j) axes, then $(X, Y, Z) = d(i, j) / \sqrt{\alpha^2 i^2 + \beta^2 j^2 + f^2} (\alpha i, \beta j, f)$, where f is the focal length of the system. These images often arise from sensor equipment that incorporates a normal intensity camera. (Fig. 22.2b.)

3. *Cylindrical*. Here, $d(i, j)$ is the distance along the line of sight of the ray through pixel (i, j) to point (X, Y, Z) . In this case, the sensor usually rotates to scan in the x direction, and translates to scan in the y direction. Thus, $(X, Y, Z) = (d(i, j) \sin(\alpha i), \beta j, d(i, j) \cos(\alpha i))$ is the usual conversion. (Fig. 22.3a.)
4. *Spherical*. Here, $d(i, j)$ is the distance along the line of sight of the ray through pixel (i, j) to point (X, Y, Z) . In this case, the sensor usually rotates to scan in the x direction, and, once each x scan, also rotates in the y direction. Thus (i, j) are the azimuth and elevation of the line of sight. Here $(X, Y, Z) = d(i, j)(\cos(\beta j) \sin(\alpha i), \sin(\beta j), \cos(\beta j) \cos(\alpha i))$. (Fig. 22.3b.)

Some sensors only record distances in a plane, so the scene (x, z) is represented by the linear image $d(i)$ for each pixel i . The orthographic, perspective, and cylindrical projection options listed above still apply in simplified form.

The second format is as a list $\{(X_i, Y_i, Z_i)\}$ of 3-D data points, but this format can be used with all of the mappings listed above. Given the conversions from image data $d(i, j)$ to (X, Y, Z) the range data is only supplied as a list. Details of the precise mapping and data format are supplied with commercial range sensors.

22.1.2 Stereo Vision

It is possible to acquire range information from many different sensors, but only a few have the reliability needed for most robotics applications. The more reliable ones, laser-based triangulation and laser radar (LIDAR), are discussed in the next section.

Real-time stereo analysis uses two or more input images to estimate the distance to points in a scene. The basic concept is *triangulation*: a scene point and the two camera points form a triangle, and knowing the baseline between the two cameras, and the angle formed by the camera rays, the distance to the object can be determined.

In practice, there are many difficulties in making a stereo imaging system that is useful for robotics applications. Most of these difficulties arise in finding reliable matches for pixels in the two images that correspond to the same point in the scene. A further consideration is that stereo analysis for robotics has a real-time constraint, and the processing power needed for some algorithms can be very high. However, in recent years much progress has been made, and the advantage of

stereo imaging is that it can provide full 3-D range images, registered with visual information, potentially out to an infinite distance, at high frame rates – something which no other range sensor can match.

In this subsection we will review the basic algorithms of stereo analysis, and highlight the problems and potential of the method. For simplicity, we use binocular stereo.

Stereo Image Geometry

This subsection gives some more detail of the fundamental geometry of stereo, and in particular the relationship of the images to the 3-D world via projection and re-projection. A more in-depth discussion of the geometry, and the rectification process, can be found in [22.2].

The input images are *rectified*, which means that the original images are modified to correspond to ideal pinhole cameras with a particular geometry, illustrated in Fig. 22.4. Any 3-D point S projects to a point in the images along a ray through the focal point. If the principal rays of the cameras are parallel, and the images are

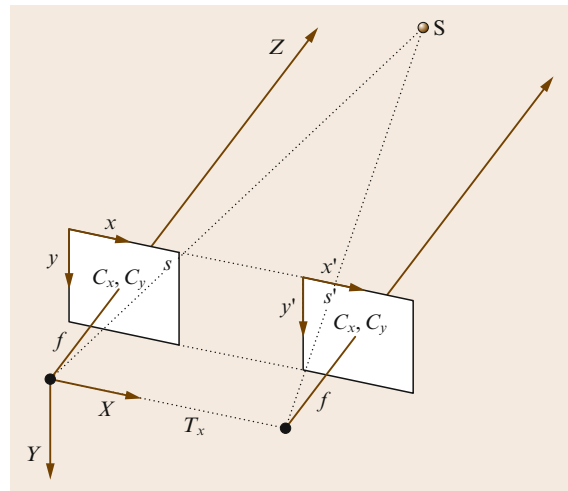


Fig. 22.4 Ideal stereo geometry. The global coordinate system is centered on the focal point (camera center) of the left camera. It is a right-handed system, with positive Z in front of the camera, and positive X to the right. The camera principal ray pierces the image plane at C_x, C_y , which is the same in both cameras (a variation for *verged* cameras allows C_x to differ between the images). The focal length is also the same. The images are lined up, with $y = y'$ for the coordinates of any scene point projected into the images. The difference between the x coordinates is called the *disparity*. The vector between the focal points is aligned with the X -axis [22.1]

embedded in a common plane and have collinear scan lines, then the search geometry takes a simple form. The *epipolar line* of a point s in the left image, defined as the possible positions of s' in the right image, is always a scan line with the same y coordinate as s . Thus, the search for a stereo match is linear. The process of finding a rectification of the original images that puts them into standard form is called *calibration*, and is discussed in [22.2].

The difference in the x coordinates of s and s' is the *disparity* of the 3-D point, which is related to its distance from the focal point, and the baseline T_x that separates the focal points.

A 3-D point can be projected into either the left or right image by a matrix multiplication in homogenous coordinates, using the *projection matrix*. The 3-D coordinates are in the frame of the left camera (Fig. 22.4).

$$P = \begin{pmatrix} F_x & 0 & C_x & -F_x T_x \\ 0 & F_y & C_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}. \quad (22.1)$$

This is the projection matrix for a single camera. F_x and F_y are the focal lengths of the rectified images, and C_x, C_y is the optical center. T_x is the translation of the camera relative to the left (reference) camera. For the left camera, it is 0; for the right camera, it is the baseline times the x focal length.

A point in three dimensions is represented by homogeneous coordinates and the projection is performed using a matrix multiplication

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = P \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}, \quad (22.2)$$

where $(x/w, y/w)$ are the idealized image coordinates.

If points in the left and right images correspond to the same scene feature, the depth of the feature can be calculated from the image coordinates using the *reprojection matrix*

$$Q = \begin{pmatrix} 1 & 0 & 0 & -C_x \\ 0 & 1 & 0 & -C_y \\ 0 & 0 & 0 & F_x \\ 0 & 0 & -1/T_x & (C_x - C_{x'})/T_x \end{pmatrix}. \quad (22.3)$$

The primed parameters are from the left projection matrix, the unprimed from the right. The last term is zero except for verged cameras. If x, y and x', y are the two

matched image points, with $d = x - x'$, then

$$\begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix} = Q \begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix}, \quad (22.4)$$

where $(X/W, Y/W, Z/W)$ are the coordinates of the scene feature, and $d = x' - x$ is the disparity. Assuming $C_x = C'_x$, the Z distance assumes the familiar inverse form of triangulation

$$Z = \frac{F - xT_x}{d}. \quad (22.5)$$

Reprojection is valid only for rectified images – for the general case, the projected lines do not intersect. The disparity d is an *inverse depth* measure, and the vector (x, y, d) is a perspective representation range image (see Sect. 22.1.1), sometimes called the *disparity space* representation. The disparity space is often used in applications instead of 3-D space, as a more efficient representation for determining obstacles or other features (see Sect. 22.3.3).

Equation (22.4) is a homography between disparity space and 3-D Euclidean space. Disparity space is also useful in translating between 3-D frames. Let $p_0 = (x_0, y_0, d_0, 1)$ in frame 0, with frame 1 related by the rigid motion R, t . From the reprojection (22.4) the 3-D position is Qp_0 . Under the rigid motion this becomes $\begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} Qp_0$, and finally applying Q^{-1} yields the disparity representation in frame 1. The concatenation of these operations is the homography

$$H(R, t) = Q^{-1} \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} Q. \quad (22.6)$$

Using the homography allows the points in the reference frame to be directly projected onto another frame, without translating to 3-D points.

Stereo Methods

The fundamental problem in stereo analysis is matching image elements that represent the same object or object part in the scene. Once the match is made, the range to the object can be computed using the image geometry.

Matching methods can be characterized as local or global. Local methods attempt to match small regions of one image to another based on intrinsic features of the region. Global methods supplement local methods by considering physical constraints such as surface continuity or base of support. Local methods can be further

classified by whether they match discrete features among images, or correlate a small area patch [22.3]. Features are usually chosen to be lighting and viewpoint independent, e.g., corners are a natural feature to use because they remain corners in almost all projections. Feature-based algorithms compensate for viewpoint changes and camera differences, and can produce rapid, robust matching, but they have the disadvantage of requiring perhaps expensive feature extraction, and yielding only sparse range results.

In the next section we present local area correlation in more detail, since it is one of the most efficient and practical algorithms for real-time stereo. A survey and results of recent stereo matching methods is in [22.5], and the authors maintain a web page listing up-to-date information [22.6].

Area Correlation Stereo

Area correlation compares small patches among images using correlation. The area size is a compromise, since small areas are more likely to be similar in images with different viewpoints, while larger areas increase the signal-to-noise ratio. In contrast to the feature-based method, area-based correlation produces dense results. Because area methods need not compute features, and have an extremely regular algorithmic structure, they can have optimized implementations.

The typical area correlation method has five steps (Fig. 22.5).

1. Geometry correction. In this step, distortions in the input images are corrected by warping into a *standard form*.
2. Image transform. A local operator transforms each pixel in the grayscale image into a more appropriate form, e.g., normalizes it based on average local intensity.
3. Area correlation. This is the correlation step, where each small area is compared with other areas in its search window.
4. Extrema extraction. The extreme value of the correlation at each pixel is determined, yielding a disparity image: each pixel value is the disparity between left and right image patches at the best match.
5. Postfiltering. One or more filters clean up noise in the disparity image result.

Correlation of image areas is disturbed by illumination, perspective, and imaging differences among images. Area correlation methods usually attempt to compensate by correlating not the raw intensity images, but some transform of the intensities. Let u, v be

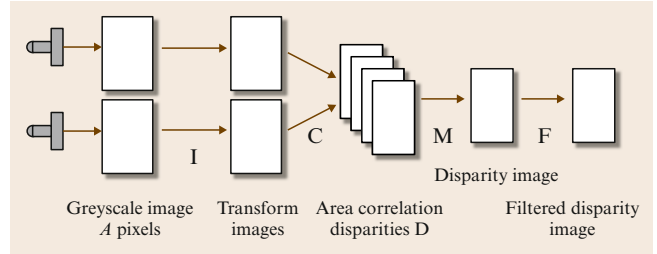


Fig. 22.5 Basic stereo processing. See text for details [22.4]

the center pixel of the correlation, x, y be pixels in the neighborhood about u, v the disparity, and $I_{x,y}, I'_{x,y}$ the intensities of the left and right images.

1. Normalized cross-correlation.

$$\frac{\sum_{x,y} [I_{x,y} - \hat{I}_{x,y}] [I'_{x-d,y} - \hat{I}'_{x-d,y}]}{\sqrt{\sum_{x,y} [I_{x,y} - \hat{I}_{x,y}]^2 \sum_{x,y} [I'_{x-d,y} - \hat{I}'_{x-d,y}]^2}}$$

2. High-pass filter such as Laplacian of Gaussian (LOG). The Laplacian measures directed edge intensities over some area smoothed by the Gaussian. Typically the standard deviation of the Gaussian is 1–2 pixels.

$$\sum_{x,y} s(\log_{x,y} - \log_{x-d,y}) .$$

Where $s(x)$ is x^2 or $||x||$ and $\log_{x,y}$ is the Laplacian of Gaussian at pixel x, y .

3. Nonparametric. These transforms are an attempt to deal with the problem of outliers, which tend to overwhelm the correlation measure, especially using a square difference. The census method [22.7] computes a bit vector describing the local environment of a pixel, and the correlation measure is the Hamming distance between two vectors

$$\sum_{x,y} (I_{x,y} > I_{u,v}) \oplus (I'_{x-d,y} > I'_{u,v}) .$$

Results on the different transforms and their error rates for some standard images are compiled in [22.6].

Another technique for increasing the signal-to-noise ratio of matches is to use more than two images [22.8]. This technique can also overcome the problem of viewpoint occlusion, where the matching part of an object does not appear in the other image. The simple technique of adding the correlations between images at the same disparity seems to work well [22.9]. Obviously, the computational expenditure for multiple images is greater than for two.

Dense range images usually contain false matches that must be filtered, although this is less of a prob-

Table 22.1 Post-filtering techniques for eliminating false matches in area correlation

Correlation surface	[22.10]
Peak width	Wide peak indicates poor feature localization
Peak height	Small peak indicates poor match
Number of peaks	Multiple peaks indicate ambiguity
Mode filter	Lack of supporting disparities violates smoothness
Left/right check [22.11, 12]	Nonsymmetric match indicates occlusion
Texture [22.13]	Low texture energy yields poor matches

lem with multiple-image methods. Table 22.1 lists some of the postfilters that have been discussed in the literature.

Disparity images can be processed to give sub-pixel accuracy, by trying to locate the correlation peak between pixels. This increases the available range resolution without much additional work. Typical accuracies are a tenth of a pixel.

Stereo Range Quality

Various artifacts and problems affect stereo range images.

Smearing. Area correlation introduces expansion in foreground objects, e.g., the woman’s head in Fig. 22.6. The cause is the dominance of strong edges on the object. Nonparametric measures are less subject to this phenomenon. Other approaches include multiple correlation windows and shaped windows.

Dropouts. These are areas where no good matches can be found because of low texture energy. Dropouts are a problem for indoor and outdoor manmade surfaces. Projecting an infrared random texture can help [22.14].

Range resolution. Unlike laser radar (LADAR) devices, stereo range accuracy is a quadratic function of distance, found by differentiating (22.5) with respect to disparity:

$$\delta Z = -\frac{F_x T_x}{d^2}.$$

(22.7)

The degradation of stereo range with distance can be clearly seen in the 3-D reconstruction of Fig. 22.6.

Processing. Area correlation is processor intensive, requiring Awd operations, where A is the image area, w the correlation window size, and d the number of disparities. Clever optimizations take advantage of redundant calculations to reduce this to Ad (independent of window size), at the expense of some

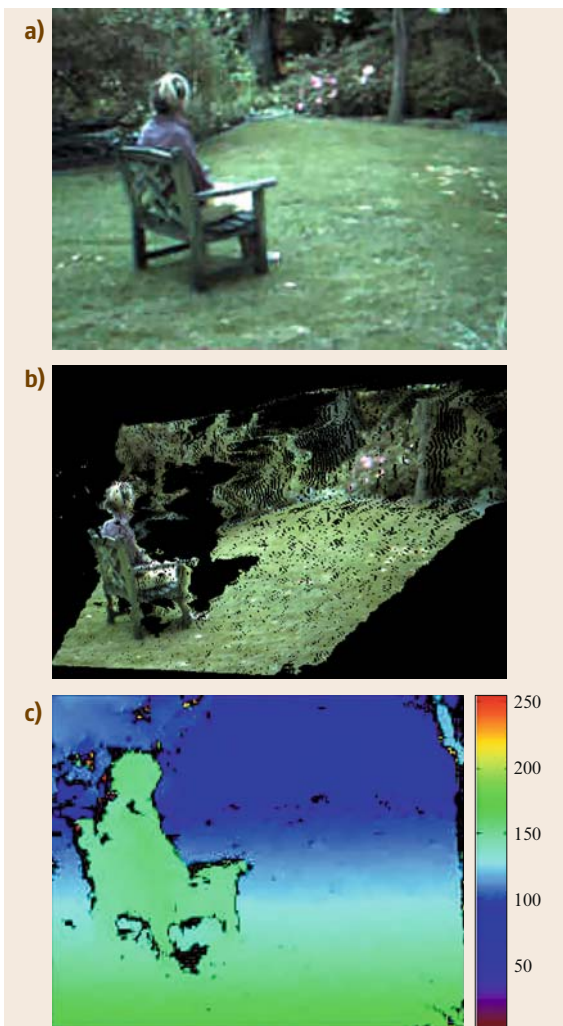


Fig. 22.6a–c Sample stereo results from an outdoor garden scene; baseline is 9 cm. (a) Original left image. (b) Computed 3-D points from a different angle. (c) Disparity in pseudo-color [22.1]

storage. Real-time implementations exist for standard personal computers (PCs) [22.1, 15], graphics accelerators [22.16, 17], digital signal processors (DSPs) [22.4], field-programmable gate arrays (FPGAs) [22.1, 18], and specialized application-specific integrated circuits (ASICs) [22.19].

Other Visual Sources of Range Information

Here we briefly list the most popular, but less reliable sources of range information. These sources can potentially supplement other sensors.

- *Focus/defocus.* Knowledge of the camera parameters and the amount of blur of image features allows estimation of how far the corresponding scene features are from the perfect focus distance [22.20]. Sensors may be passive (using a precaptured image) or active (capturing several images with different focus settings).
- *Structure and motion.* Structure and motion algorithms compute 3-D scene structure and the sensor positions simultaneously [22.21]. This is essentially a binocular stereo process (see discussion before), except that only a single moving camera is used. Thus the images needed by the stereo process are acquired by the same camera in several different positions. Video camcorders are also used, but have lower resolution. One important advantage of this approach over the normal algorithm is that features can be tracked easily if the time between frames or the motion is small enough. This simplifies the *correspondence problem*, however it can lead to another problem. If the pair of images used for the stereo calculation are taken close together in time, then the separation between the cameras images will not be much, that is, there will be a *short baseline*. Triangulation calculations are then more inaccurate, as small errors in estimating the position of image features results in large errors in the estimated 3-D position (particularly the depth estimate). This problem can be partly avoided by tracking for longer periods. A second problem that can arise is that not all motions are suitable for estimating the full 3-D scene structure, for example, if the video recorder only rotates about its optical axis or about its focus point, then no 3-D information can be recovered. A little care can avoid this problem.
- *Shading.* The pattern of shading on a surface is related to the orientation of the surface relative to the observer and light sources. This relationship can be used to estimate the surface orientation across the surface. The surface normals can then be integrated to give an estimate of the relative surface depth.
- *Photometric stereo.* Photometric stereo [22.22] is a combination of shading and stereo processes. The key concept is that the shading of an object varies with the position of the light sources. Hence, if you had several aligned pictures of an object or scene with the light source in different positions (e.g., the sun moved), then you can calculate the scene's surface normals. From these, the relative surface depth can be estimated. The restriction of a stationary observer and changing light sources makes this

approach less likely to be useful to most robotics applications.

- *Texture.* The way uniform or statistical textures vary on a surface is related to the orientation of the surface relative to the observer. As with shading, the texture gradients can be used to estimate the surface orientation across the surface [22.23]. The surface normals can then be integrated to give an estimate of the relative surface depth.

22.1.3 Laser-Based Range Sensors

There are three types of laser-based range sensors in common use:

1. triangulation sensors
2. phase-modulation sensors
3. time-of-flight sensors

These are discussed in more detail below. There are also Doppler and interference laser-based range sensors, but these seem to not be in general use at the moment so we skip them here. An excellent recent review of range sensors is by *Blais* [22.24].

The physical principles behind the three types of sensors discussed below do not intrinsically depend on the use of a laser – for the most part any light source would work. However, lasers are traditional because: (1) they can easily generate bright beams with lightweight sources, (2) infrared beams can be used unobtrusively, (3) they focus well to give narrow beams, (4) single-frequency sources allow easier rejection filtering of unwanted frequencies, (5) single-frequency sources do not disperse from refraction as much as full-spectrum sources, (6) semiconductor devices can more easily generate short pulses, etc.

One advantage of all three sensor types is that it is often possible to acquire a reflectance image registered with the range image. By measuring the amount that the laser beam strength has reduced after reflection from the target, one can estimate the reflectance of the surface. This is only the reflectance at the single spectral frequency of the laser, but, nonetheless, this gives useful information about the appearance of the surface (as well as the shape as given by the range measurement); Three-color laser systems [22.25] similarly give registered red–green–blue (RGB) color images. Figure 22.1 shows registered range and reflectance images from the same scene.

One disadvantage of all three sensor types is specular reflections. The normal assumption is that the observed light is a diffuse reflection from the surface. If the ob-

served surface is specular, such as polished metal or water, then the source illumination may be reflected in unpredictable directions. If any light eventually is detected by the receiver, then it is likely to cause incorrect range measurements. Specular reflections are also likely at the fold edges of surfaces.

A second problem is the laser *footprint*. Because the laser beam has a finite width, when it strikes the edge of a surface, part of the beam may be actually lie on a more distant surface. The consequences of this will depend on the actual sensor, but it commonly results in range measurements that lie between the closer and more distant surfaces. These measurements thus suggest that there is surface where there really is empty space. Some noise removal is possible in obvious empty space, but there may be erroneous measurements that are so close to the true surface that they are difficult to eliminate.

22.1.4 Time-of-Flight Range Sensors

Time-of-flight range sensors are exactly that: they compute distance by measuring the time that a pulse of light takes to travel from the source to the observed target and then to the detector (usually collocated with the source). In a sense, they are radar sensors that are based on light. The travel time multiplied by the speed of light (in the given medium – space, air or water – and adjusted for the density and temperature of the medium) gives the distance. Laser-based time-of-flight range sensors are also called light detection and ranging (LIDAR) or laser radar (LADAR) sensors.

Limitations on the accuracy of these sensors are based on the minimum observation time (and thus the minimum distance observable), the temporal accuracy (or quantization) of the receiver, and the temporal width of the laser pulse.

Many time-of-flight sensors used for local measurements have what is called an *ambiguity interval*, for example, 20 m. The sensor emits pulses of light periodically, and computes an average target distance from the time of the returning pulses. To limit noise from reflections and simplify the detection electronics, many sensors only accept signals that arrive within time Δt , but this time window might also observe previous pulses reflected by more distant surfaces. This means that a measurement Z is ambiguous to the multiple of $1/2c\Delta t$ because surfaces that are further away (e.g., z) than $1/2c\Delta t$ are recorded as $z \bmod 1/2c\Delta t$. Thus distances on smooth surfaces can increase until they reach $c\Delta t/2$ and then they become 0. Typical values for $c\Delta t/2$ are 20–40 m. On smooth surfaces, such as a ground

plane or road, an unwinding algorithm can recover the true depth by assuming that the distances should be changing smoothly.

Most time-of-flight sensors transmit only a single beam, thus range measurements are only obtained from a single surface point. Robotics applications usually need more information, so the range data is usually supplied as a vector of range to surfaces lying in a plane (Fig. 22.7) or as an image (Fig. 22.1). To obtain these denser representations, the laser beam is swept across the scene. Normally the beam is swept by a set of mirrors rather than moving the laser and detector themselves (mirrors are lighter and less prone to motion damage). The most common technologies for this are using a stepper motor (for program-based range sensing) or rotating or oscillating mirrors for automatic scanning.

Typical ground-based time-of-flight sensors suitable for robotics applications have a range of 10–100 m, and an accuracy of 5–10 mm. The amount of the scene scanned will depend on the sweep rate of the mirrors and the pulse rate, but 1–25 K points per second are typical. Manufacturers of these sensors include Acuity, SICK, Mensi, DeltaSphere, and CyraX.

Recently, a type of time-of-flight range sensor called the *flash LADAR* has been developed. The key innovation has been the inclusion of timing circuits at each pixel of the sensor chip. Thus, each pixel can measure the time at which a light pulse is observed from the line of sight viewed by that pixel. This allows simultaneous calculation of the range values at each pixel. The light pulse now has to cover to whole portion of the scene that is observed, so sensors typically use an array of infrared laser light-emitting diodes (LEDs). While spatial resolution is smaller than current cameras (e.g., 64×64 , 160×124 , 128×128), the data can be acquired at video rates (30–50 fps), which provides considerable informa-

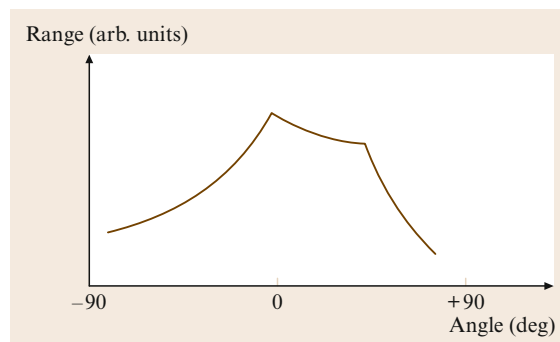


Fig. 22.7 Plot of ideal one-dimensional range image of sample distance versus angle of measurement

tion usable for robot feedback. Different sensor ranges have been reported, such as up to 5 m [22.26] (with on the order of 5–50 cm standard deviation depending on target distance and orientation) or at 1.1 km [22.27] (no standard deviation reported).

22.1.5 Modulation Range Sensors

Modulation-based range sensors are commonly of two types, where a continuous laser signal is either amplitude or frequency modulated. By observing the phase shift between the outgoing and return signals, the signal transit time is estimated and from this the target distance. As the signal phase repeats every 2π , these sensors also have an ambiguity interval.

These sensors also produce a single beam that must be swept. Scan ranges are typically 20–40 m and the accuracy is 5 mm. Figure 22.1 was captured with a modulation sensor.

22.1.6 Triangulation Range Sensors

Triangulation range sensors [22.28] are based on principles similar to the stereo sensors discussed previously. The key concept is illustrated in Fig. 22.8: a laser beam is projected from one position onto the observed surface. The light spot that this creates is observed by a sensor from a second position. Knowing the relative positions and orientations of the laser and sensor plus some simple trigonometry allows calculation of the 3-D position of the illuminated surface point. The triangulation process is more accurate when the laser spot's position is accurately estimated. About 0.1 pixel accuracy can be normally achieved [22.29].

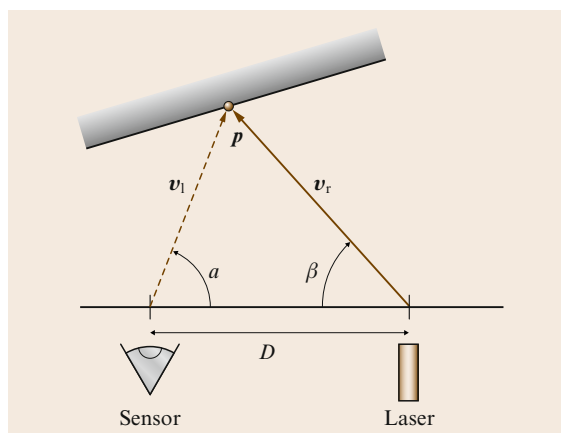


Fig. 22.8 Triangulation using a laser spot

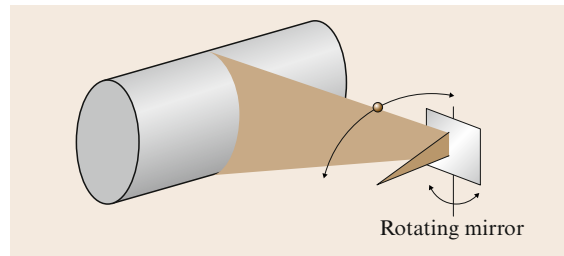


Fig. 22.9 A swept laser plane covers a larger scene

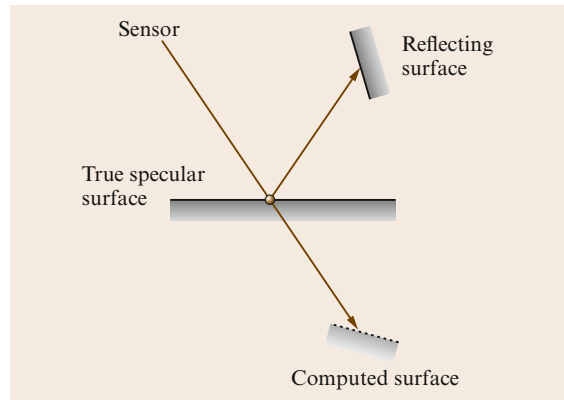


Fig. 22.10 Incorrect estimated depths on specular surfaces

Because the laser illumination can be controlled, this gives a number of practical advantages.

1. A laser of known frequency (e.g., 733 nm) can be matched with a very selective spectral filter of the same frequency (e.g., 5 nm half-width). This normally allows unique identification of the light spot as the filtering virtually eliminates all other bright light, leaving the laser spot as the brightest spot in the image.
2. The laser spot can be reshaped with lenses or mirrors to create multiple spots or stripes, thus allowing the measurement of multiple 3-D points simultaneously. Stripes are commonly used because these can be swept across the scene (see Fig. 22.9 for an example) to observe a full scene. Other illumination patterns are also commonly used, such as parallel lines, concentric circles, cross hairs, and dot grids. Commercial structured light pattern generators are available from, e.g., Lasiris or Edmunds Optics.
3. The laser light ray can be positioned by mirrors under computer control to selectively observe a given area, such as around a doorway or potential obstacle, or an object that might be grasped.

Disadvantages include

1. potential eye safety risks from the power of lasers, particularly when invisible laser frequencies are used (commonly infrared),
2. specular reflections from metallic and polished objects, which may cause incorrect calculation of where the laser spot is illuminating, as in Fig. 22.10, where the hypothesized surface lies behind the true surface.

22.1.7 Example Sensors

A typical phase-modulation-based range sensor is the Zohler and Frohlich 25200. This is an expensive spherical scan sensor, observing a full 360° horizontally and 155° vertically. Each scan can acquire up to 20 000 3-D points horizontally and 8000 points vertically in about 100 s. The accuracy of the scan depends on the distance to the target, but 4 mm is typical, with samples every 0.01° (samples every 1.7 mm at 10 m distance). The densely sensed data is typically used for 3-D scene survey, modeling, and virtual-reality reconstruction.

An example of a laser sensor that has been commonly used for robot navigation is the SICK LMS 200.

This sensor sweeps a laser ray in a horizontal plane through an arc of 180° , acquiring 720 range measurements in a plane in 0.05 s. While a plane might seem limiting, the observed 3-D data can be easily matched to a stored (or previously learned) model of the environment at the height of the scan. This allows accurate estimation of the sensor's position in the scene (and thus a mobile vehicle that carries the sensor). Of course, this requires an environment without overhanging structures that are not detectable at the scan height. Another common configuration is to mount the sensor high on the vehicle with the scan plane tilted downwards. In this configuration the ground in front of the vehicle is progressively scanned as the vehicle moves forward, thus allowing detection of potential obstacles.

The Minolta 910 is a small triangulation scanner with an accuracy of about 0.01 mm in a range of up to 2 m, for about 500^2 points in 2.5 s. It has commonly been used for small-region scanning, such as for inspection or part modeling, but can also be mounted in a fixed location to scan a robot workcell. Observation of the workcell allows a variety of actions, such as inspection of parts, location of dropped or bin delivered components, or accurate part position determination.

More examples and details of range sensors for robot navigation are presented in Sect. 22.3.

22.2 Registration

This section introduces techniques for 3-D localization of parts for robot manipulation, self-localization of robot vehicles, and scene understanding for robot navigation. All of these are based on the ability to register 3-D shapes, e.g., range images to range images, triangulated surfaces or geometric models. Before we introduce these applications in Sect. 22.2.7, we first look at some basic techniques for manipulating 3-D data.

22.2.1 3-D Feature Representations

There are many representations available for encoding 3-D scene structure and model representations, but the following representations are the ones most commonly encountered in robotics applications. Some scene models or descriptions may use more than one of these simultaneously to describe different aspects of the scene or object models.

3-D Point Sets

This is a set $\{\mathbf{p}_i = (X_i, Y_i, Z_i)\}$ of 3-D points that describe some salient and identifiable points in the scene. They might be the centers of spheres (often used as markers), corners where three planes intersect, or the extrema of some protrusion or indentation on a surface. They may be a subset of an initially acquired 3-D full scene point set, they might be extracted from a range image, or they might be computed theoretical points based on extracted data features.

Planes

This is a set of planar surfaces observed in a scene or bounding part or all of an object model. There are several ways to represent a plane, but one common representation is by the equation of the plane $ax + by + cz + d = 0$ that passes through corresponding scene or model surface points $\{\mathbf{p}_i = (X_i, Y_i, Z_i)\}$. A plane only needs three parameters, so there are usually

some additional constraints on the plane representation (a, b, c, d) , such as $d = 1$ or $a^2 + b^2 + c^2 = 1$. The vector $\mathbf{n} = (a, b, c)'/a^2 + b^2 + c^2$ is the surface normal.

A planar surface may only be described by the infinite surface as given by the equation, but it may also include a description of the boundary of the surface patch. Again there are many possible representations for curves, either in three or two dimensions lying in the plane of the surface. Convenient representations for robotics applications are lists of the 3-D points $\{(X_i, Y_i, Z_i)\}$ that form the patch boundary, or polylines, which represent the boundary by a set of connected line segments. A polyline can be represented by the sequence of 3-D points $\{(X_i, Y_i, Z_i)\}$ that form the vertices that join the line segments.

Triangulated Surfaces

This representation describes an object or scene by a set of triangular patches. More general polygonal surface patches or even various smooth surface representations are also used, but triangles are most commonly used because they are simpler and there are inexpensive PC graphics cards that display triangles at high speed.

The triangles can be large (e.g., when representing planar surfaces) or small (e.g., when representing curved surfaces). The size chosen for the triangles reflects the accuracy desired for representing the object or scene sur-

faces. The triangulated surface might be complete in the sense that all observable scene or objects surfaces are represented by triangles, or there might be disconnected surface patches with or without internal holes. For grasping or navigation, you do not want any unrepresented scene surface to lie in front of the represented portion of the surface where a gripper or vehicle might collide with it. Hence, we assume that the triangulation algorithms produce patch sets that, if completely connected at the edges, implicitly bound all real scene surfaces. Figure 22.11 shows an example of a triangulated surface over the original underlying smooth surface.

There are many slightly different schemes for representing the triangles [22.30], but the main features are a list $\{\mathbf{v}_i = (X_i, Y_i, Z_i)\}$ indexed by i of the 3-D vertices at the corners of the triangles, and a list of the triangles $\{t_n = (i_n, j_n, k_n)\}$ specifying the indices of the corner vertices.

From the triangular surface patches, one can compute potential grasping positions, potential navigable surfaces or obstacles, or match observed surface shapes to previously stored shapes.

3-D Lines

The 3-D lines where planar surfaces meet are features that can be easily detected by both stereo and range sensors. These features occur commonly in built environments (e.g., where walls, floors, ceilings, and doorways meet, around the edges of wall structures like notice boards, at the edges of office and warehouse furniture, etc.). They are also common on manmade objects. In the case of stereo imaging, changes of surface shape or coloring are detected as edges, which can be matched in the stereo process to directly produce the 3-D edge. In the case of a range sensor, planar surfaces can be easily extracted from the range data (see next section) and adjacent planar surfaces can be intersected to give the edges.

The most straightforward representation for 3-D lines is the set of points $\mathbf{x} = \mathbf{p} + \lambda \mathbf{v}$ for all λ , where \mathbf{v} is a unit vector. This has five degrees of freedom; more complex representations, e.g., with four degrees of freedom exist [22.2].

Voxels

The voxel (volume pixel) approach represents the 3-D world by 3-D boxes/cells that indicate where there is scene structure and where there is free space. The simplest representation is a 3-D binary array, encoded as 1 for having structure and 0 for free space. This can be quite memory intensive, and also requires a lot of

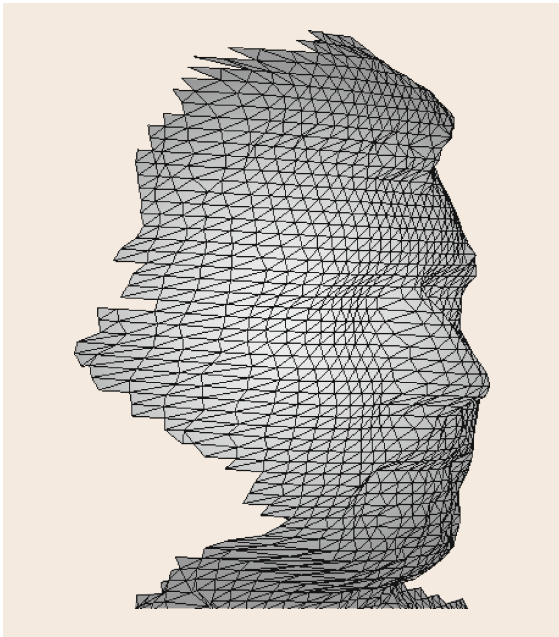


Fig. 22.11 Triangulated surface (thanks to T. Breckon)

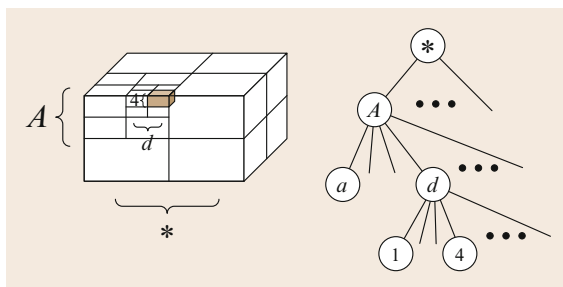


Fig. 22.12 Recursive space decomposition and a portion of the corresponding octree

computation to check many voxels for content. A more complex but more compact representation is the hierarchical representation called the octree [22.30]. This divides the entire (bounded) rectangular space into eight rectangular subspaces called octants (Fig. 22.12). A tree data structure encodes the content of each octant as empty, full or mixed. Mixed octants are then subdivided into eight smaller rectangular octants, encoded as subtrees of the larger tree. Subdivision continues until some minimum octant size is reached. Determining whether a voxel is empty, full or mixed depends on the sensor used, however, if no 3-D data points are located in the volume of a voxel, then it is likely to be empty. Similarly, if many 3-D points are present, then the voxel is likely to be full. The remaining voxels can be considered to be mixed. More details of a voxelization algorithm can be found in [22.31].

For the purpose of robot navigation, localization or grasping, only the surface and free-space voxels need to be marked accurately. The interior of objects and scene structure are largely irrelevant.

22.2.2 3-D Feature Extraction

Three types of structured 3-D features are of particular interest for robot applications: straight lines, planes, and triangulated surfaces. How these are extracted from range data sets is summarized here:

Planes

Planes are generally found by some region-growing process, based on a seed patch. When the scene largely consists of planes, a particularly simple approach is based on selecting a previously unused point and the set of points $\{v_i = (X_i, Y_i, Z_i)\}$ in its neighborhood. A plane is fit to these points using a least-squares method: create a $4 \times N$ matrix D of the N points in the set, each point augmented with a 1. Call these ex-

tended points $\{v_i\}$. As each point lies in the same plane $ax + by + cz + d = 0$, we can easily estimate the plane parameter vector $p = (a, b, c, d)'$ as follows. We know that a perfect plane fit to perfect data would satisfy $Dp = 0$. Since there is noise, we instead create the error vector $e = Dp$ and find the vector p that minimizes its length (squared) $e \cdot e = p'D'Dp$. This vector is the eigenvector with the smallest eigenvalue of the matrix $D'D$. This initial hypothesized plane then needs to be tested for reasonableness by (1) examining the smallest eigenvalue, which should be small and on the order of the square of the expected noise level, and (2) ensuring that most of the 3-D points in the fitted set lie on the plane (i.e., $|v_i \cdot p| < \tau$).

Larger planar regions are *grown* by locating new adjacent points v_i that lie on the plane (i.e., $|v_i \cdot p| < \tau$). When enough of these are found, the parameters p of the plane are re-estimated. Points on the detected plane are removed and the process is repeated with a new seed patch. This process continues until no more points can be added. More complete descriptions of planar feature extraction can be found in [22.32].

When the planes are more sparsely found in the scene, then another approach is the random sample consensus (RANSAC) feature detection algorithm [22.33]. When adapted for plane finding, this algorithm selects three 3-D points at random (although exploiting some locality to the point selection algorithm can improve the efficiency of the algorithm). These three points determine a plane with parameter vector p . All points $\{v_i\}$ in the set are testing for belonging to the plane (i.e., $|v_i \cdot p| < \tau$). If enough points are close to the plane, then potentially a plane has been found. These points should also be processed to find a connected set, from which a more accurate set of plane parameters can be estimated using the least-squares algorithm given above. If a planar patch is successfully found, the points that lie in that plane are removed from the dataset. The random selection of three points then continues until no more planes are found (a bound on how many tries to make can be estimated).

Straight Lines

While straight lines are common in manmade scenes, direct extraction from 3-D datasets is not easy. The main source of the difficulty is that 3-D sensors often do not acquire good responses at edges of surfaces as (1) the sensor beam will be sampling from two different surfaces and 2) laser-based sensors produce somewhat unpredictable effects. For this reason, most 3-D line detection algorithms are indirect, whereby planes are first

detected, e.g., using the method of the previous section, and then adjacent planes are intersected. Adjacency can be tested by finding paths of connected pixels that lead from one plane to the other. If planes 1 and 2 contains points p_1 and p_2 and have surface normals n_1 and n_2 , respectively, then the resulting intersection line has equation $x = a + \lambda d$, where a is a point on the line and $d = n_1 \times n_2 / \|n_1 \times n_2\|$ is the line direction. There are an infinite number of possible points a , which can be found by solving the equations $a' n_1 = p'_1 n_1$ and $a' n_2 = p'_2 n_2$. A reasonable third constraint that obtains a point near to p_2 is the equation $a' d = p'_2 d$. This gives us an infinite line. Most practical applications require a finite segment. The endpoints can be estimated by (1) finding the points on the line that lie close to observed points in both planes and then (2) finding the two extremes of those points. On the other hand, finding straight 3-D lines can be easier with a stereo sensor, as these result from matching two straight two-dimensional (2-D) image lines.

Triangulated Surfaces

The goal of the process is to estimate a triangulated mesh from a set of 3-D data points. If the points are sampled on a regular grid (e.g., from a $2\frac{1}{2}$ -D range image), then the triangles can be formed naturally from the sampling (Fig. 22.13). If the points are part of a 3-D point set, then triangulation is harder. We do not give details here because of the complexity of this task, but some of the issues that have to be considered are: (1) how to find a surface that lies close to all data points (as it is unlikely to actually pass through all because of noise), (2) how to avoid holes and wild triangles in the

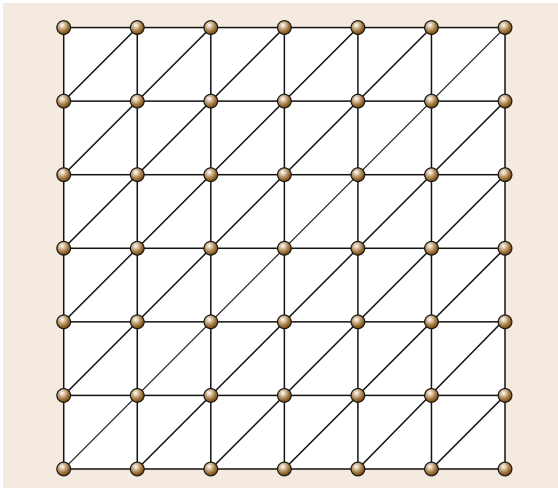


Fig. 22.13 Triangulation of a regular grid of points

mesh, (3) how to choose the threshold of closeness so that only one surface passes through sections of a point cloud, (4) how to choose the triangle size, (5) what to do about outlying points, and (6) how to maintain observable scene features, such as surface edges. Popular early approaches for triangulation are the marching cube and triangle [22.34, 35] algorithms. These algorithms can give meshes with many triangles. The triangle size and thus the computational complexity of using the mesh can be reduced by mesh decimation, for which many algorithms exist [22.36, 37].

22.2.3 Model Matching and Multiple-View Registration

Model matching is the process of matching some stored representation to some observed data. In the case discussed here, we assume that both are 3-D representations. Furthermore, we assume that the representations being matched are both of the same type, e.g., 3-D model and scene lines. (While different types of data can also be matched, we ignore these more specialized algorithms here.)

A special case of matching is when the two structures being matched are both scene or model surfaces. This commonly occurs when attempting to produce a larger representation of the scene by fusing multiple partial views of the scene. These partial views can be acquired as a mobile vehicle navigates around the scene, observing it from different places. The mobile robotics simultaneous localization and mapping (SLAM) algorithm [22.38] (discussed in Chap. 37) incrementally fuses newly observed scene structure to its growing scene map model, while also matching portions of that newly observed data to previously observed data to estimate its current location. One SLAM project [22.39] used scale invariant feature transform (SIFT) feature points for 3-D modeling and map construction.

The algorithm used for matching depends on the complexity the structures being matched. If the structures being matched are extended geometric entities such as planes or 3-D lines, then a discrete matching algorithm like the interpretation tree algorithm [22.40] can be used. Alternatively, if the structures are low level, such as 3-D points or triangles, then a point set alignment algorithm like the iterated closest point (ICP) algorithm [22.41] can be used. Both of these are described in more detail below.

After structures have been matched, the commonest next step is to estimate the transformation (rotation plus translation) linking the two datasets. This process is described in the next section. With the transforma-

tion, we can then transform the data into the same coordinate system and fuse (combine) the datasets. If the structures being fused are points, then the result is a larger point set formed from the combination of the two (or more) original datasets. Sets of planes and lines can also be simply added together, subject to representing the matched members of the two datasets only once. Triangulated surfaces require a more complex merging, in that we require a topologically correct triangulated mesh to be the result. The zippering algorithm [22.42] is a well-known triangulated mesh merging algorithm.

The interpretation tree algorithm [22.40] (see Table 22.2) is suitable for matching small numbers (e.g., less than about 20–30) of discrete objects, such as vertical edges seen in 2-D or 3-D. If there are M model and D data objects, then potentially there are M^D different matches. The key to efficient matching is to identify pairwise constraints that eliminate unsuitable matches. Constraints between pairs of model features and pairs of data features also greatly reduce the matching space. If the constraints eliminate enough features, a polynomial-time algorithm results. The core of the algorithm is defined as follows. Let $\{m_i\}$ and $\{d_j\}$ be the sets of model and data features to be matched, $u(m_i, d_j)$ is true if m_i and d_j are compatible features, $b(m_i, m_j, d_k, d_l)$ is true if the four model and data features are compatible, and T is the minimum number of matched features before a successful match is declared. Pairs is the set of successfully matched features. The function `true sizeof` counts the number of actual matches in the set, disregarding

matches with the wildcard ‘*’ which matches anything.

When the data being matched are sets of points or triangles, then the ICP algorithm [22.41] is more commonly used. This algorithm estimates the pose transformation that aligns the two sets to a minimum distance. With this transformation, the two sets can be represented in the same coordinate system and treated as a larger single set (perhaps with some merging of overlapping data).

Here we give the algorithm for point matching, but this can be easily adapted for other sets of features. This algorithm is iterative, so may not always converge quickly, but often several iterations is sufficient. However, ICP can produce a bad alignment of the data sets. Best results arise with a good initial alignment, such as from odometry or previous positions. The ICP algorithm can be extended to include other properties when matching, such as color or local neighborhood structure. A good spatial indexing algorithm (e.g., k-d trees) is necessary for efficient matching in the closest point function (CP) below.

Let \mathcal{S} be a set of N_s points $\{s_1, \dots, s_{N_s}\}$ and \mathcal{M} be the model. Let $\|s - m\|$ be the Euclidean distance between point $s \in \mathcal{S}$ and $m \in \mathcal{M}$. Let $\text{CP}(s, \mathcal{M})$ be the closest point (Euclidean distance) in \mathcal{M} to the scene point s .

1. Let $T^{[0]}$ be an initial estimate of the rigid transformation aligning the two sets.
2. Repeat for $k = 1 \dots k_{\max}$ or until convergence:
 - a) Compute the set of correspondences $\mathcal{C} = \bigcup_{i=1}^{N_s} \{(s_i, \text{CP}(T^{[k-1]}(s_i), \mathcal{M}))\}$.

Table 22.2 Interpretation tree algorithm

```

pairs=it(0,{})
if true sizeof(pairs) >= T, then success

function pairs=it(level,inpairs)
  if M-level+true sizeof(inpairs) < T
    then return {} % can never succeed
  for each d_i % loopD start
    if not u(m_level,d_i), then continue loopD
    for each (m_k,d_l) in inpairs
      if not b(m_level,m_k,d_i,d_l)
        then continue loopD
    endfor % have found a successful new pair to add
    pairs = it(level+1, union(inpairs,(m_level,d_i)))
    if true sizeof(pairs) >= T, then return pairs
  endfor % loopD end
% no success, so try wildcard
pairs = it(level+1,union(inpairs,(m_level,*)))

```

- b) Compute the new Euclidean transformation $T^{[k]}$ that minimizes the mean square error between point pairs in \mathcal{C} .

22.2.4 Maximum-Likelihood Registration

One popular technique for range scan matching, especially in the 2-D case, uses maximum likelihood as the criterion for match goodness. This technique also takes into account a probabilistic model of the range readings. Let r be a range reading of a sensor scan at position s , and \bar{r} the distance to the nearest object along the path of r (which depends on s). Then a model of the sensor reading is the probability $p(r|\bar{r})$. Typically the model has a Gaussian shape around the correct range, with an earlier false-positive region and a missed-reading region at the end (Fig. 22.14).

A scan is matched when its position produces a maximum-likelihood result for all the readings, based on a reference scan. Assuming independence of the readings, from Bayes' rule we get

$$\max_s p(s|\mathbf{r}) = \max_s \prod_i p(r_i|\bar{r}_i).$$

The maximum likelihood scan location can be found by hill-climbing or exhaustive search from a good starting point. Figure 22.15 shows the results of maximum likelihood for aligning a scan against several reference scans. In this 2-D case, the reference scans are put into an occupancy grid for computing \bar{r} [22.44]. A widespread modification for efficiency is to compute \bar{r} by a smearing operation on the occupancy grid, ignoring line-of-sight

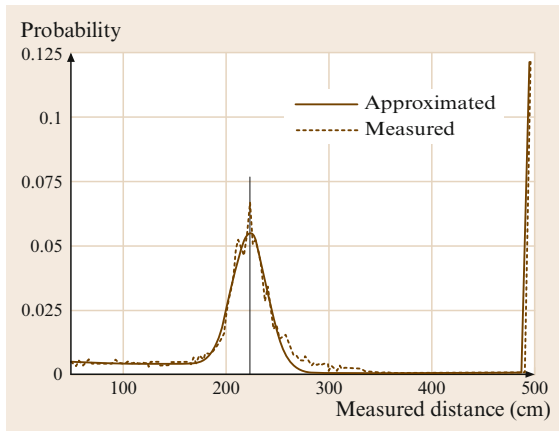


Fig. 22.14 Probability profile $p(r|\bar{r})$ for a laser sensor reading. The Gaussian peak occurs at the distance \bar{r} of the obstacle [22.43]



Fig. 22.15 Scan match by maximum likelihood against several reference scans in an occupancy grid [22.44]

information [22.45]. In the 3-D case, triangulated surfaces are more appropriate than voxels [22.46].

22.2.5 Multiple-Scan Registration

In the previous subsection, multiple reference scans could be used to form an occupancy grid or surface triangles for matching. In general, scan matching between sets of scans produces a network of constraints among the scans, for example, in going around a loop, successive scans form a chain of constraints, and the first and last scans form a closed loop of constraints. Globally consistent registration of scans is a part of **SLAM** (Chap. 37).

If the individual constraints have covariance estimates, then maximum-likelihood techniques can be used to find a globally consistent estimate for all the scans [22.47]. This global registration is among the scan *poses*, and does not involve the scans themselves – all information has been abstracted to constraints among the poses. Let \bar{s}_{ij} be the scan-matched pose difference between scans s_i and s_j , with covariance Γ_{ij} . The maximum-likelihood estimate for all s is given by the nonlinear least-squares system

$$\min_s \sum_{ij} (s_{ij} - \bar{s}_{ij})^\top \Gamma_{ij} (s_{ij} - \bar{s}_{ij}), \quad (22.8)$$

which can be solved efficiently using iterative least-squares techniques, e.g., Levenberg–Marquadt or conjugate gradient methods [22.48, 49].

A complication is that the system of constraints is calculated based on an initial set of poses s . Given a repositioning of s from (22.8), redoing the scan matching will give a different set of constraints, in general. Iterating the registration process with new constraints is not guaranteed to lead to a global minimum; in practice, with a good initial estimate, convergence is often rapid and robust.

22.2.6 Relative Pose Estimation

Central to many tasks is the estimation of the coordinate system relative position or pose transformation between

two coordinate systems, for example, this might be the pose of a scanner mounted on a mobile vehicle relative to scene landmarks, or it might be the relative pose of some scene features as observed in two views taken from different positions.

We present here three algorithms that cover most instances of the pose estimation process, which differ slightly based on the type of feature being matched.

Point Set Relative Pose Estimation

The first algorithm matches point sets. Assume that N 3-D points $\{m_i\}$ in one structure or coordinate system are matched to points $\{d_i\}$ in a different coordinate system. The matching might be found as part of the alignment process in the ICP algorithm just described or they might be observed triangle vertices matched to a previously constructed triangulated scene model, for example. The desired pose is the rotation matrix R and translation vector t that minimizes $\sum_i \|\mathbf{R}m_i + t - d_i\|^2$. Compute the mean vectors μ_m and μ_d of the two sets. Compute the centralized point sets by $a_i = m_i - \mu_m$ and $b_i = d_i - \mu_d$. Construct the $3 \times N$ matrix A that consists of the vectors $\{a_i\}$ stacked up. Construct the $3 \times N$ matrices B in a similar way from the vectors $\{b_i\}$. Compute the singular value decomposition $\text{svd}(BA') = U'DV'$ [22.50]. Compute the rotation matrix $R = VU'$. If the data points are nearly planar then this calculation can introduce a mirror-image transformation. This can be checked with the vector triple product and, if mirrored, the diagonal correction matrix $C(1,1,-1)$ can be used in $R = VC U'$. Finally, compute the translation vector $t = \mu_d - R\mu_m$. With these least-squares estimates of the transformation, a point m_i transforms to point $Rm_i + t$, which should be near to point d_i .

Straight Line Relative Pose Estimation

If 3-D lines are the features that are extracted, then the relative pose transformation can be estimated as follows. Assume N paired lines. The first set of lines is described by direction vectors $\{e_i\}$ and a point on each line $\{a_i\}$. The second set of lines is described by direction vectors $\{f_i\}$ and a point on each line $\{b_i\}$. In this algorithm, we assume that the direction vectors on the matched segments always point the same direction (i.e., they are not inverted). This can be achieved by exploiting some scene constraints, or trying all combinations and eliminating inconsistent solutions. The points a_i and b_i need not correspond to the same point after alignment. The desired rotation matrix R minimizes $\sum_i \|\mathbf{R}e_i - f_i\|^2$. Construct the $3 \times N$ matrices E that consist of the vectors $\{e_i\}$ stacked up. Construct the $3 \times N$ matrices F in a sim-

ilar way from the vectors $\{f_i\}$. Compute the singular value decomposition $\text{svd}(FE') = U'DV'$. Compute the rotation matrix $R = VU'$. The translation estimate t minimizes the sum of the square of the distances λ_i between the rotated points a_i and corresponding line (f_i, b_i) . Define the matrix $L = \sum_i (I - f_i f_i')(I - f_i f_i')$. Define the vector $n = \sum_i (I - f_i f_i')(I - f_i f_i')(Ra_i - b_i)$. Then the translation is $t = -L^{-1}n$.

Plane Relative Pose Estimation

Finally, if planes are the 3-D features extracted for matching, then the relative pose transformation can be estimated as follows. Assume N paired planes. The first set of planes is described by surface normals $\{e_i\}$ and a point on each plane $\{a_i\}$. The second set of planes is described by surface normals $\{f_i\}$ and a point on each plane $\{b_i\}$. Here we assume that the surface normals always point outward from the surface. The points a_i and b_i need not correspond to the same point after alignment. The desired rotation matrix R minimizes $\sum_i \|\mathbf{R}e_i - f_i\|^2$. Construct the $3 \times N$ matrices E that consist of the vectors $\{e_i\}$ stacked up. Construct the $3 \times N$ matrices F in a similar way from the vectors $\{f_i\}$. Compute the singular value decomposition $\text{svd}(FE') = U'DV'$ [22.50]. Compute the rotation matrix $R = VU'$. The translation estimate t minimizes the sum of the square of the distances λ_i between the rotated point a_i and the corresponding plane (f_i, b_i) . Define the matrix $L = \sum_i f_i f_i'$. Define the vector $n = \sum_i f_i f_i'(Ra_i - b_i)$. Then the translation is $t = -L^{-1}n$.

In all of the calculations described above, we assumed normally distributed errors. For techniques to make these sorts of calculations more robust, see Zhang [22.51].

22.2.7 3-D Applications

This section links the techniques presented above to the robotics applications of 3-D localization of parts for robot manipulation, self-localization of robot vehicles, and scene understanding for robot navigation. The robotics tasks mentioned here are discussed in more detail in other chapters in the series. While this chapter focuses on robotics applications, there are many other 3-D sensing applications. An area of much current research is that of acquiring 3-D models, particularly for reverse engineering of mechanical parts [22.52], historical artifacts [22.53], buildings [22.54], and people for computer games and movies (see, e.g., Cyberware's Whole-Body X 3-D scanner).

The key tasks in robot manipulation are: (1) identification of grasping points (Chaps. 27 and 28),

(2) identification of a collision-free grasp (Chaps. 27 and 28), (3) recognition of parts to be manipulated (Chap. 23) and (4) position estimation of parts for manipulation (Chaps. 23 and 42).

The key tasks in robot navigation and self-localization are: (5) identification of a navigable ground plane (Sect. 22.3), (6) identification of a collision-free path (Chap. 35), (7) identification of landmarks (Chap. 36), and (8) estimation of vehicle location (Chaps. 37 and 40).

The mobile and assembly robotics tasks link together rather naturally. Tasks 1 and 5 have a connection, when we consider these tasks in the context of unknown parts or paths. Part grasping requires finding regions on a part that are graspable, which usually means locally planar patches that are large enough that a gripper can make good contact with them. Similarly, navigation usually requires smooth ground regions that are large enough for the vehicle – again locally planar patches. Both tasks are commonly based on triangulated scene methods to rep-

resent the data, from which connected regions of nearly coplanar patches can be extracted. The main difference between these two tasks is the ground-plane detection task is looking for a larger patch, that must be on the *ground* and upward facing.

Tasks 2 and 6 require a method of representing empty space along the proposed trajectory of the gripper contacts to the vehicle. The voxel representation is good for this task.

Tasks 3 and 7 are model matching tasks and can use the methods of Sect. 22.2.3 to match observed scene features to prestored models of known parts or scene locations. Commonly used features are large planar surfaces, 3-D edges, and 3-D feature points.

Tasks 4 and 8 are pose estimation tasks and can use the methods of Sect. 22.2.6 to estimate the pose of the object relative to the sensor or vehicle (i. e., sensor) relative to the scene. Again, commonly used features are large planar surfaces, 3-D edges, and 3-D feature points.

22.3 Navigation and Terrain Classification

One of the more compelling uses for range data is for navigation of mobile robot vehicles. Range data provides information about obstacles and free space for the vehicle, in a direct geometric form. Because of the real-time constraints of navigation, it is often impractical to reconstruct a full 3-D model of the terrain using the techniques presented in this Chapter. Instead, most systems use an *elevation model*. An elevation model is a tessellated 2-D representation of space, where at each cell there is information about the distribution of 3-D points in the cell. In its simplest incarnation, the elevation map just contains the mean height of range points above the nominal ground plane (Fig. 22.16). This representation is sufficient for some indoor and urban environments; more sophisticated versions that determine a local plane, scatter of points in the cell, etc., are useful for more complicated off-road driving. Elevation maps marked with obstacles have obvious utility for planning a collision-free path for the vehicle.

22.3.1 Indoor Reconstruction

SLAM algorithms (Chap. 37) using 2-D laser range finders can reconstruct floor plans with centimeter precision. Some research has extended this work to 3-D reconstruc-

tion, using 2-D lasers that are swept along as the robot moves [22.56]. The resultant point cloud is typically registered using the pose of the robot as corrected by the 2-D **SLAM** algorithm, rather than any of the 3-D registration techniques covered in this chapter, because the laser is swept using robot motion.

The raw points can be presented as a 3-D image, or triangulated to give a planar or mesh reconstruction of indoor surfaces. The latter is especially compelling when camera images are texture-mapped onto the surfaces, creating a realistic 3-D model. Figure 22.17 shows some results from indoor mapping using this technique.

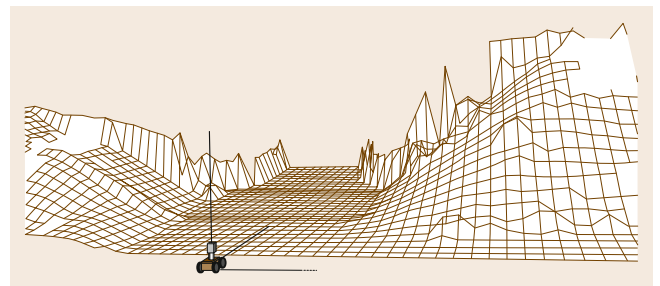


Fig. 22.16 Elevation map in urban terrain. Each cell holds the height of the terrain at that point. More extensive features can also be incorporated: slope, point variance, etc. [22.55]



Fig. 22.17 3-D indoor map from a swept vertical plane LADAR. Registration is from a horizontal LADAR using SLAM algorithms [22.56]

Smoothing of surface facets can be used to recover planar surfaces [22.57].

22.3.2 Urban Navigation

In urban navigation, the environment is structured, with roads, buildings, sidewalks, and also moving objects – people and other vehicles. There are two main challenges: how to register laser scans from a fast-moving vehicle for consistent mapping, and how to detect moving objects using range scans (of course, other methods are also used for detecting moving objects, e.g., appearance-based vision).

Outdoor vehicles can use precision global positioning system (GPS), inertial measurement units, and wheel odometry to keep track of their position and orientation, typically with an extended Kalman filter. This method is good enough to obviate the need for precise registration matching among scans, as long as the motion model of

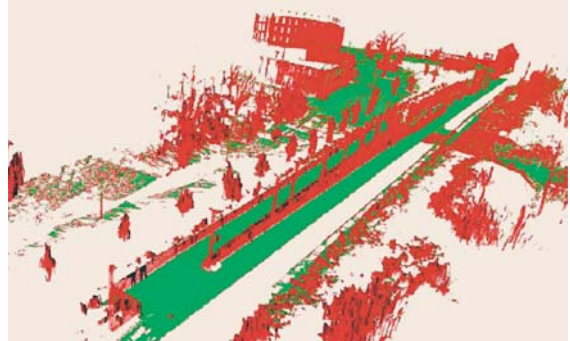


Fig. 22.18 Elevation map of an urban scene, using $10\text{ cm} \times 10\text{ cm}$ cells. Obstacles in red, ground plane in green [22.55]

the vehicle, and timing from the range scanner, is used to place each scan reading in its proper position in the world model. This method also works in relatively easy off-road terrain such as in the Defense Advanced Research Projects Agency (DARPA) Grand Challenge [22.59]. In all cases, the reduction of pose estimation error is critical for good performance [22.60].

Once scan readings are registered using the vehicle pose estimation, they can be put into an elevation map, and obstacles detected using the slope and vertical extent of the range readings in the cells of the map. A complication is that there may be multiple levels of elevation in an urban setting, for example, an overpass would not be an obstacle if it were high enough. One proposal is to use multiple elevation clusters within each cell; this technique is called a *multilevel surface map* (MLS, [22.55]). Each cell in the map stores a set of surfaces represented by a mean height and variance. Figure 22.18 shows an MLS with a cell size of 10 cm^2 , with ground plane and obstacles marked.

For dynamic objects, real-time stereo at 15–30 Hz can capture the motion of the objects. When the stereo rig is fixed, range background subtraction isolates just



Fig. 22.19a–c Independent motion detection from a moving platform. The reference image (a) is forward-projected using the motion homography to (b); (c) is the difference with the actual image [22.58]

the moving objects [22.61]. When the rig is on a moving vehicle, the problem is more difficult, since the whole scene is moving with respect to the rig. It can be solved by estimating the motion of the rig with respect to the dominant rigid background of the scene. Let R, t be the motion of the rig between two frames, estimated by extracting features and matching them across the two temporal frames, using the techniques of Chap. 37. The homography $H(R, t)$ of (22.6) provides a direct projection of the disparity vectors $p_0 = (x_0, y_0, d_0, 1)$ of the first frame to their correspondences $H(R, t)p_0$ under R, t in the second frame. Using the homography allows the points in the reference frame to be directly projected onto the next frame, without translating to 3-D points. Figure 22.19 shows the projected pixels under rigid motion from a reference scene. The difference between the projected and actual pixels gives the independently moving objects [22.58].

22.3.3 Rough Terrain

Rough outdoor terrain presents two challenges:

- There may be no extensive ground plane to characterize driveability and obstacles.
- Vegetation that is pliable and driveable may appear as an obstacle in range images.

Figure 22.20 shows a typical outdoor scene, with a small (1 m) robot driving through vegetation and rough ground [22.62]. Range data from stereo vision on the robot will see the top of the vegetation and some ground points below. The elevation model can be extended to look at *point statistics* within each cell, to capture the notion of a local ground plane and penetrability related to vegetation. In [22.65], for example, the set of proposed features includes:

- major plane slope using a robust fit (Sect. 22.2.2)
- height difference of maximum and minimum heights
- points above the major plane
- density: the ratio of points in the cell to rays that pass through the cell

The density feature is interesting (and expensive to compute), and attempts to characterize vegetation such as grass or bushes, by looking at whether range readings penetrate an elevation cell. The idea of using range readings to estimate vegetation has been discussed in several other projects on off-road driving [22.64, 66, 67].

Elevation map cells can be characterized as obstacles or driveable through learning or hand-built classifiers. Among the learning techniques are neu-



Fig. 22.20 Rough terrain, no ground plane, driveable vegetation [22.62]

ral nets [22.65] and Gaussian mixture models with expectation-maximization learning [22.63]. The latter work also includes a lower level of interpretation, classifying surfaces into planar patches (ground plane, solid obstacles), linear features (telephone wires), and scattered features (vegetation). Figure 22.21 shows some results from a laser-scanned outdoor scene. Linear features such as telephone wires and the telephone pole are accurately determined, as well as vegetation with high penetrability.

Some additional problems occur in rough-terrain navigation. For planar laser range finders that are swept over the terrain by vehicle motion, the precision of vehicle pose estimation is important for accurate reconstruction. Attitude errors of less than 0.5° can cause false positives in obstacle detection, especially for sweeps far



Fig. 22.21 Classification using point statistics, from [22.63]. Red is planar surface, blue is thin linear surface, green is scattered penetrable surface [22.64]

ahead of the vehicle. In [22.60], this problem is solved by looking at the time of each laser reading, and noting a correlation between height errors and time difference in the readings.

Negative obstacles (ditches and cliffs) are difficult to detect with range information, because the sensor may not see the bottom of the obstacle. This is espe-

cially true for vehicle-mounted sensors that are not very high off the ground, and that are looking far ahead. Negative obstacles can be inferred when there is a gap in the ground plane, and a plane slanted upwards at the back edge of the gap. Such artifacts can be efficiently found using column search on the disparity image [22.68].

22.4 Conclusions and Further Reading

Range sensing is an active and expanding field of research in robotics. The presence of new types of devices – flash **LADARs**, multi-beam **LADARs**, on-camera stereo processing – and the continuing development of robust algorithms for object reconstruction, localization, and mapping has helped to bring applications out of the laboratory and into the real world. Indoor navigation with **LADARs** is already being exploited in commercial products (see, for example, [22.69]). As the basic capabilities become more robust, researchers are looking to perform useful tasks, such as fetching items or doing dishes [22.70].

Another set of challenges are found in less benign environments, such as urban and off-road driving (**DARPA** Grand Challenge and Urban Challenge [22.59]). Stereo vision and laser range finding also will play a role in helping to provide autonomy for a new generation of more capable robotic platforms that rely on walking for locomotion [22.71]. The challenges are dealing with motion that is less smooth than wheeled platforms, environments that contain dynamic obstacles, and task-oriented recognition of objects.

References

- 22.1 Videre Design LLC: www.videredesign.com, accessed Nov 12, 2007 (Videre Design, Menlo Park 2007)
- 22.2 R. Hartley, A. Zisserman: *Multiple view geometry in computer vision* (Cambridge Univ. Press, Cambridge 2000)
- 22.3 S. Barnard, M. Fischler: Computational stereo, *ACM Comput. Surv.* **14**(4), 553–572 (1982)
- 22.4 K. Konolige: Small vision system. hardware and implementation, *Proc. Int. Symp. Robot. Res.* (Hayama 1997) pp. 111–116
- 22.5 D. Scharstein, R. Szeliski, R. Zabih: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, *Int. J. Comput. Vis.* **47**(1/2/3), 7–42 (2002)
- 22.6 D. Scharstein, R. Szeliski: Middlebury College Stereo Vision Research Page, vision.middlebury.edu/stereo, accessed Nov 12, 2007 (Middlebury College, Middlebury 2007)
- 22.7 R. Zabih, J. Woodfill: Non-parametric local transforms for computing visual correspondence, *Proc. Eur. Conf. on Computer Vision*, Vol. 2 (Stockholm 1994) pp. 151–158
- 22.8 O. Faugeras, B. Hotz, H. Mathieu, T. Viéville, Z. Zhang, P. Fua, E. Théron, L. Moll, G. Berry, J. Vuillemin, P. Bertin, C. Proy: Real time correlation based stereo: algorithm implementations and applications, Tech. Report RR-2013, INRIA (1993)
- 22.9 M. Okutomi, T. Kanade: A multiple-baseline stereo, *IEEE Trans. Patt. Anal. Mach. Intell.* **15**(4), 353–363 (1993)
- 22.10 L. Matthies: Stereo vision for planetary rovers: stochastic modeling to near realtime implementation, *Int. J. Comput. Vis.* **8**(1), 71–91 (1993)
- 22.11 R. Bolles, J. Woodfill: Spatiotemporal consistency checking of passive range data, *Proc. Int. Symp. on Robotics Research* (Hidden Valley 1993)
- 22.12 P. Fua: A parallel stereo algorithm that produces dense depth maps and preserves image features, *Mach. Vis. Appl.* **6**(1), 35–49 (1993)
- 22.13 H. Moravec: Visual mapping by a robot rover, *Proc. Int. Joint Conf. on AI (IJCAI)* (Tokyo 1979) pp. 598–600
- 22.14 A. Adan, F. Molina, L. Morena: Disordered patterns projection for 3D motion recovering, *Proc. Int. Conf. on 3D Data Processing, Visualization and Transmission* (Thessaloniki 2004) pp. 262–269
- 22.15 Point Grey Research Inc.: www.ptgrey.com, accessed Nov 12, 2007 (Point Grey Research, Vancouver 2007)
- 22.16 C. Zach, A. Klaus, M. Hadwiger, K. Karner: Accurate dense stereo reconstruction using graphics

- hardware, Proc. EUROGRAPHICS (Granada 2003) pp. 227–234
- 22.17 R. Yang, M. Pollefeys: Multi-resolution real-time stereo on commodity graphics hardware, Int. Conf. Computer Vision and Pattern Recognition, Vol. 1 (Madison 2003) pp. 211–217
- 22.18 Focus Robotics Inc.: www.focusrobotics.com, accessed Nov 12, 2007 (Focus Robotics, Hudson 2007)
- 22.19 TYZX Inc.: www.tyzx.com, accessed Nov 12, 2007 (TYZX, Menlo Park 2007)
- 22.20 S.K. Nayar, Y. Nakagawa: Shape from focus, IEEE Trans. Patt. Anal. Mach. Intell. **16**(8), 824–831 (1994)
- 22.21 M. Pollefeys, R. Koch, L. Van Gool: Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters, Int. J. Comput. Vis. **32**(1), 7–25 (1999)
- 22.22 A. Hertzmann, S.M. Seitz: Example-based photometric stereo: Shape reconstruction with general, varying BRDFs, IEEE Trans. Patt. Anal. Mach. Intell. **27**(8), 1254–1264 (2005)
- 22.23 A. Lobay, D.A. Forsyth: Shape from texture without boundaries, Int. J. Comput. Vis. **67**(1), 71–91 (2006)
- 22.24 F. Blais: Review of 20 years of range sensor development, J. Electron. Imag. **13**(1), 231–240 (2004)
- 22.25 R. Baribeau, M. Rioux, G. Godin: Color reflectance modeling using a polychromatic laser range sensor, IEEE Trans. Patt. Anal. Mach. Intell. **14**(2), 263–269 (1992)
- 22.26 D. Anderson, H. Herman, A. Kelly: Experimental Characterization of Commercial Flash Ladar Devices, Int. Conf. of Sensing and Technology (Palmerston North 2005) pp. 17–23
- 22.27 R. Stettner, H. Bailey, S. Silverman: Three-Dimensional Flash Ladar Focal Planes and Time-Dependent Imaging, Advanced Scientific Concepts, 2006; Technical Report (February 23, 2007): [www.advancedscientificconcepts.com/images/Three Dimensional Flash Ladar Focal Planes-ISSSR Paper.pdf](http://www.advancedscientificconcepts.com/images/Three%20Dimensional%20Flash%20Ladar%20Focal%20Planes-ISSSR%20Paper.pdf), accessed Nov 12, 2007 (Advanced Scientific Concepts, Santa Barbara 2007)
- 22.28 J.J. LeMoigne, A.M. Waxman: Structured light patterns for robot mobility, Robot. Autom. **4**, 541–548 (1988)
- 22.29 R.B. Fisher, D.K. Naidu: A Comparison of Algorithms for Subpixel Peak Detection. In: *Image Technology*, ed. by J. Sanz (Springer, Berlin, Heidelberg 1996)
- 22.30 J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes: *Computer Graphics: principles and practice* (Addison Wesley, Reading 1996)
- 22.31 B. Curless, M. Levoy: A Volumetric Method for Building Complex Models from Range Images, Proc. of Int. Conf. on Comput. Graph. and Inter. Tech. (SIGGRAPH) (New Orleans 1996) pp. 303–312
- 22.32 A. Hoover, G. Jean-Baptiste, X. Jiang, P.J. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, R. Fisher: An experimental comparison of range segmentation algorithms, IEEE Trans. Patt. Anal. Mach. Intell. **18**(7), 673–689 (1996)
- 22.33 M.A. Fischler, R.C. Bolles: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM **24**(6), 381–395 (1981)
- 22.34 H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, W. Stuetzle: Surface reconstruction from unorganized points, Comput. Graph. **26**(2), 71–78 (1992)
- 22.35 A. Hilton, A. Stoddart, J. Illingworth, T. Windeatt: Implicit surface-based geometric fusion, Comput. Vis. Image Under. **69**(3), 273–291 (1998)
- 22.36 H. Hoppe: New quadric metric for simplifying meshes with appearance attributes, IEEE Visualization 1999 Conference (San Francisco 1999) pp. 59–66
- 22.37 W.J. Schroeder, J.A. Zarge, W.E. Lorensen: Decimation of triangle meshes, Proc. of Int. Conf. on Comput. Graph. and Inter. Tech. (SIGGRAPH) (Chicago 1992) pp. 65–70
- 22.38 S. Thrun: A probabilistic online mapping algorithm for teams of mobile robots, Int. J. Robot. Res. **20**(5), 335–363 (2001)
- 22.39 J. Little, S. Se, D. Lowe: Vision based mobile robot localization and mapping using scale-invariant features, Proc. IEEE Inf. Conf. on Robotics and Automation (Seoul 2001) pp. 2051–2058
- 22.40 E. Grimson: *Object Recognition by Computer: The Role of Geometric Constraints* (MIT Press, London 1990)
- 22.41 P.J. Besl, N.D. McKay: A method for registration of 3D shapes, IEEE Trans. Patt. Anal. Mach. Intell. **14**(2), 239–256 (1992)
- 22.42 G. Turk, M. Levoy: Zippered Polygon Meshes from Range Images, Proc. of Int. Conf. on Comput. Graph. and Inter. Tech. (SIGGRAPH) (Orlando 1994) pp. 311–318
- 22.43 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
- 22.44 D. Haehnel, D. Schulz, W. Burgard: Mapping with mobile robots in populated environments, Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Vol. 1 (Lausanne 2002) pp. 496–501
- 22.45 K. Konolige, K. Chou: Markov localization using correlation, Proc. Int. Joint Conf. on AI (IJCAI) (Stockholm 1999) pp. 1154–1159
- 22.46 D. Haehnel, W. Burgard: Probabilistic Matching for 3D Scan Registration, Proc. of the VDI-Conference Robotik 2002 (Robotik) (Ludwigsburg 2002)
- 22.47 F. Lu, E. Milios: Globally consistent range scan alignment for environment mapping, Auton. Robot. **4**, 333–349 (1997)
- 22.48 K. Konolige: Large-scale map-making, Proceedings of the National Conference on AI (AAAI) (San Jose 2004) pp. 457–463
- 22.49 A. Kelly, R. Unnikrishnan: Efficient Construction of Globally Consistent Ladar Maps using Pose Network

- Topology and Nonlinear Programming, Proc. Int. Symp of Robotics Research (Siena 2003)
- 22.50 K.S. Arun, T.S. Huang, S.D. Blostein: Least-squares fitting of two 3-D point sets, *IEEE Trans. Patt. Anal. Mach. Intell.* **9**(5), 698–700 (1987)
- 22.51 Z. Zhang: Parameter estimation techniques: a tutorial with application to conic fitting, *Image Vis. Comput.* **15**, 59–76 (1997)
- 22.52 P. Benko, G. Kos, T. Varady, L. Andor, R.R. Martin: Constrained fitting in reverse engineering, *Comput. Aided Geom. Des.* **19**, 173–205 (2002)
- 22.53 M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, D. Fulk: The Digital Michelangelo Project: 3D Scanning of Large Statues, Proc. 27th Conf. on Computer graphics and interactive techniques (SIGGRAPH) (New Orleans 2000) pp. 131–144
- 22.54 I. Stamos, P. Allen: 3-D Model Construction Using Range and Image Data, Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Vol. 1 (Hilton Head Island 2000) pp. 531–536
- 22.55 R. Triebel, P. Pfaff, W. Burgard: Multi-level surface maps for outdoor terrain mapping and loop closing, Proc. of the IEEE Int. Conf. on Intel. Robots and Systems (IROS) (Beijing 2006)
- 22.56 S. Thrun, W. Burgard, D. Fox: A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping, Proc. IEEE Inf. Conf. on Robotics and Automation (San Francisco 2000) pp. 321–328
- 22.57 Y. Liu, R. Emery, D. Chakrabarti, W. Burgard, S. Thrun: Using EM to Learn 3D Models of Indoor Environments with Mobile Robots, Proc. Int. Conf. on Machine Learning (Williamstown 2001) pp. 329–336
- 22.58 M. Agrawal, K. Konolige, L. Iocchi: Real-time detection of independent motion using stereo, IEEE Workshop on Motion (Breckenridge 2005) pp. 207–214
- 22.59 The DARPA Grand Challenge: www.darpa.mil/grandchallenge05, accessed Nov 12, 2007 (DARPA, Arlington 2005)
- 22.60 S. Thrun, M. Montemerlo, H. Dahlkamp et al.: Stanley: The robot that won the DARPA Grand Challenge, *J. Field Robot.* **23**(9), 661–670 (2006)
- 22.61 C. Eveland, K. Konolige, R. Bolles: Background modeling for segmentation of video-rate stereo sequences, Proc. Int. Conf. on Computer Vision and Pattern Recog (Santa Barbara 1998) pp. 266–271
- 22.62 K. Konolige, M. Agrawal, R.C. Bolles, C. Cowan, M. Fischler, B. Gerkey: Outdoor mapping and Navigation using Stereo Vision, Intl. Symp. on Experimental Robotics (ISER) (Rio de Janeiro 2006)
- 22.63 J. Lalonde, N. Vandapel, D. Huber, M. Hebert: Natural terrain classification using three-dimensional lidar data for ground robot mobility, *J. Field Robot.* **23**(10), 839–861 (2006)
- 22.64 J.-F. Lalonde, N. Vandapel, M. Hebert: Data structure for efficient processing in 3-D, *Robotics: Science and Systems 1*, Cambridge (2005)
- 22.65 M. Happold, M. Ollis, N. Johnson: enhancing supervised terrain classification with predictive unsupervised learning, *Robotics: Science and Systems* (Philadelphia 2006)
- 22.66 R. Manduchi, A. Castano, A. Talukder, L. Matthies: Obstacle detection and terrain classification for autonomous off-road navigation, *Auton. Robot.* **18**, 81–102 (2005)
- 22.67 A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Happold, H. Herman, R. Mandelbaum, T. Pilarski, P. Rander, S. Thayer, N. Vallidis, R. Warner: Toward reliable off road autonomous vehicles operating in challenging environments, *Int. J. Robot. Res.* **25**(5–6), 449–483 (2006)
- 22.68 P. Bellutta, R. Manduchi, L. Matthies, K. Owens, A. Rankin: Terrain Perception for Demo III, Proc. of the 2000 IEEE Intelligent Vehicles Conf. (Dearborn 2000) pp. 326–331
- 22.69 KARTO: Software for robots on the move. www.kartorobotics.com, accessed Nov 12, 2007 (ISRI, Menlo Park 2007)
- 22.70 The Stanford Artificial Intelligence Robot: www.cs.stanford.edu/group/stair, accessed Nov 12, 2007 (Stanford Univ., Stanford 2007)
- 22.71 Perception for Humanoid Robots. www.ri.cmu.edu/projects/project_595.html, accessed Nov 12, 2007 (Carnegie Mellon Univ., Pittsburgh 2007)

3-D Vision and Recognition

Kostas Daniilidis, Jan-Olof Eklundh

In this chapter, we describe methods to be applied on a robot equipped with one or more camera sensors. Our goal is to present representations and models for both three-dimensional (3-D) motion and structure estimation as well as recognition. We do not delve into estimation and inference issues since these are extensively treated in other chapters. The same applies to the fusion with other sensors, which we heavily encourage but do not describe here.

In the first part we describe the main methods in 3-D inference from two-dimensional (2-D) images. We are at the point where we could propose a recipe, at least for a small spatial extent. If we are able to track a few visual features in our images, we are able to estimate the self-motion of the robot as well as its pose with respect to any known landmark. Having solutions for minimal case problems, the obvious way here is to apply random sample consensus. If no known 3-D landmark is given then the trajectory of the camera exhibits drift. From the trajectory of the camera, time windows over several frames are selected and a 3-D dense depth map is obtained through solving the stereo problem. Large-scale reconstructions based on camera only do raise challenges with respect to drift and loop closing.

In the second part we deal with recognition as appealed to robotics. The main challenge here is to detect an instance of an object and recognize or categorize it. Since in robotics applications an

23.1 3-D Vision and Visual SLAM	544
23.1.1 Pose Estimation Solution	545
23.1.2 Triangulation	545
23.1.3 Moving Stereo	546
23.1.4 Structure from Motion (SfM)	547
23.1.5 Monocular SLAM or Multiple-View SfM	548
23.1.6 Dense Depth Maps from Stereo	549
23.2 Recognition	551
23.2.1 Approaches to Recognition	551
23.2.2 Appearance-Based Methods	552
23.2.3 Matching	555
23.2.4 Constellation-Based Methods – Recognition by Parts	557
23.2.5 Place Recognition and Terrain Classification	558
23.3 Conclusion and Further Reading	558
References	559

object of interest always resides in a cluttered environment any algorithm has to be insensitive to missing parts of the object of interest and outliers. The dominant paradigm is based on matching the appearance of pictures. Features are detected and quantized into visual words. Similarity is based on the difference between histograms of such visual words. Recognition has a long way to go but robotics provides the opportunity to explore an object and be active in the recognition process.

With the rapid progress and cost reduction in digital imaging, cameras became the standard and probably the cheapest sensor on a robot. Unlike positioning (global position system, **GPS**), inertial measurement unit (**IMU**), and distance sensors (sonar, laser, infrared) cameras produce the highest bandwidth of data. One video camera of very modest resolution yields a bandwidth of

140 Mbits/s (= 30 frames/s × (640 × 480) pixels/frame × 16 bits/pixel). Exploiting information useful for a robot from such a bit stream is less explicit than in the case of **GPS** or a laser scanner but semantically richer.

Assume for example the scenario that a robot vehicle is given the task of going from place A to place B given as instruction only intermediate visual landmarks and/or

GPS waypoints. The robot starts at A and has to decide where is a drivable path. Such a decision can be accomplished through the detection of obstacles from at least two images by estimating a depth or occupancy map with a *stereo* algorithm. While driving, the robot wants to estimate its trajectory which can be accomplished with a *structure-from-motion* algorithm. The result of the trajectory can be used to build a layout of the environment through *matching* and *triangulation*, which in turn can be used as a reference for a subsequent *pose estimation*. At each time instance the robot has to parse the surrounding environment for risks like pedestrians,

or for objects it is searching for like a trash can. It has to become aware of *loop closing* or a reentry if the robot has been kidnapped or blinded for a while. This can be accomplished through *object and scene recognition* yielding the *what* and *where* of objects around the robot. In an extreme scenario, a vehicle can be left to explore a city and build a semantic 3-D map as well as a trajectory of all places it visited, the ultimate *visual simultaneous localization and semantic mapping* problem. In the next section we will deal with 3-D motion and estimation and mapping and in the last section with object recognition.

23.1 3-D Vision and Visual SLAM

In this section, we are going to treat all the above problems assuming that we have only one or two cameras capturing visible light. Though many of those problems can be considerably simplified if we assume the existence of active sensors (for example, based on structured light projection) or if we restrict the configuration space of a robot, we are not going to make use of any such assumptions. Instead we want to increase the understanding of a reasonably general setting and leave to the user the fusion with estimates from other sensors or the reduction of the space of unknowns. The reader is referred to Chaps. 21, 22, and 36 regarding range sensors and to Chap. 20 regarding GPS and inertial sensors. In this chapter we mainly present models, leaving the probabilistic estimation to the chapters about sensor fusion (Chap. 25) and SLAM (Chap. 37). Throughout this section we assume that the correspondence between point features has been solved. The matching problem and the detection of features and their descriptors is treated in the recognition session.

Let us start by introducing the projection of the world to an image plane. Assume that a point in the world (X, Y, Z) has coordinates (X_{ci}, Y_{ci}, Z_{ci}) with respect to the coordinate system of a camera c_i related to each other by the following transformation

$$\begin{pmatrix} X_{ci} \\ Y_{ci} \\ Z_{ci} \end{pmatrix} = \mathbf{R}_i \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + \mathbf{t}_i, \quad (23.1)$$

where \mathbf{R}_i is a rotation matrix whose columns are the world axes with respect to the camera. The translation vector \mathbf{t}_i is starting from the origin of the camera and

ending at the origin of the world coordinate system. The rotation matrix is orthogonal $\mathbf{R}^T \mathbf{R} = \mathbf{I}$, with determinant 1. We assume that the center of projection is the origin of the coordinate system and that the optical axis is the Z_{ci} -axis of the camera. If we assume that the image plane is the plane $Z_{ci} = 1$ then the image coordinates (x_i, y_i) read

$$x_i = \frac{X_{ci}}{Z_{ci}} \quad y_i = \frac{Y_{ci}}{Z_{ci}}. \quad (23.2)$$

In practice, what we measure are the pixel coordinates (u_i, v_i) in the image, which are related to the image coordinates (x_i, y_i) by the affine transformation

$$u_i = f\alpha x_i + \beta y_i + c_u \quad v_i = fy_i + c_v, \quad (23.3)$$

where f is the distance of the image plane to the projection center measured in pixels. It is also called the focal length, because they are considered approximately equal. The aspect ratio α is a scaling induced by non-square sensor cells or different sampling rates horizontally and vertically. The skew factor β accounts for a shearing induced by a non-perfectly frontal image plane. The image center c_u, c_v is the point of intersection of the image plane with the optical axis. These five parameters are called intrinsic parameters and the process of recovering them is called intrinsic calibration. Upon recovering them we can talk about a calibrated system and we can work with the image coordinates (x_i, y_i) instead of the pixel coordinates (u_i, v_i) . In many vision systems, in particular on mobile robots, wide-angle lenses introduce a radial distortion around the image

center which can be modeled polynomially:

$$\begin{aligned} x_i^{\text{dist}} &= x_i(1 + k_1 r + k_2 r^2 + k_3 r^3 + \dots), \\ y_i^{\text{dist}} &= y_i(1 + k_1 r + k_2 r^2 + k_3 r^3 + \dots), \\ \text{where } r^2 &= x_i^2 + y_i^2, \end{aligned}$$

where we temporarily assumed that the image center is at (0,0). The image coordinates (x_i, y_i) in (23.3) have to be replaced with the distorted coordinates $(x_{\text{dist}}, y_{\text{dist}})$.

Recovering the intrinsic parameters when we can make multiple views of a reference pattern like a checkerboard without variation of the intrinsic parameters has become a standard procedure using tools like the MATLAB calibration toolbox or Zhang's OpenCV calibration function [23.1]. When intrinsics like the focal length vary during operation and viewing reference patterns is not practically feasible, we rely on the state-of-the-art method by Pollefeys et al. [23.2, 3]. When all intrinsics are unknown we can use the Kruppa equations and several stratified self-calibration approaches [23.4, 5] that require at least three views. Apart from radial distortion, the projection relations shown above can be summarized in matrix form. By denoting $\mathbf{u}_i = (u_i, v_i, 1)$ and $\mathbf{X} = (X, Y, Z, 1)$ we obtain

$$\lambda_i \mathbf{u}_i = \mathbf{K}_i (\mathbf{R}_i \quad \mathbf{t}_i) \mathbf{X} = \mathbf{P} \mathbf{X}, \quad (23.4)$$

where $\lambda_i = Z_{ci}$ is the depth of point \mathbf{X} in camera coordinates and \mathbf{P} is the 3×4 projection matrix. The depth λ_i can be eliminated to obtain two equations relating the world to the pixel coordinates.

23.1.1 Pose Estimation Solution

When we have landmarks in the world with known positions \mathbf{X} , and we can measure their projections, the problem of recovering the unknown rotation and translation in the calibrated case is called pose estimation. Of course, this presumes the identification of the world landmarks in the image. In robotics, pose estimation is rather known as a variant of localization in a known environment. We assume that a camera is calibrated and that measurements of N points are given in world coordinates $\mathbf{X}_{j=1, \dots, N}$ and calibrated image coordinates $\mathbf{x}_{j=1, \dots, N}$. Let us assume two scene points and denote the known angle between their projections \mathbf{x}_1 and \mathbf{x}_2 as δ_{12} (Fig. 23.1). Let us denote the squared distance $\|\mathbf{X}_i - \mathbf{X}_j\|^2$ by d_{ij}^2 and the lengths of \mathbf{X}_j by d_j^2 . Then the cosine law reads

$$d_1^2 + d_2^2 - 2d_1 d_2 \cos \delta_{12} = d_{12}^2. \quad (23.5)$$

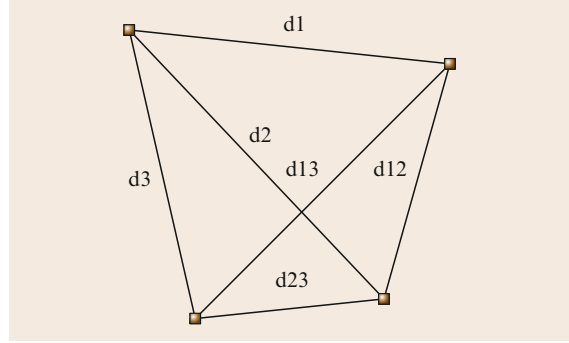


Fig. 23.1 Pose estimation problem: a camera seeing three points at unknown distances d_1 , d_2 , and d_3 with known angles between the rays and known point distances d_{12} , d_{13} , and d_{23}

If we can recover d_1 and d_2 the rest will be an absolute orientation problem

$$d_j \mathbf{x}_j = \mathbf{R} \mathbf{X}_j + \mathbf{t} \quad (23.6)$$

to recover the translation and rotation between the camera and the world coordinate system.

The cosine law has two unknowns d_1 and d_2 so with three points we should be able to solve for the pose estimation problem. Indeed, three points yield a system of three quadratic equations in three unknowns, with a maximum of eight solutions.

We follow here the analysis of the classic solution in [23.6] and set $d_2 = u d_1$ and $d_3 = v d_1$ and solve all three equations for d_1 :

$$\begin{aligned} d_1 &= \frac{d_{23}^2}{u^2 + v^2 - 2uv \cos \delta_{23}}, \\ d_1 &= \frac{d_{13}^2}{1 + v^2 - 2v \cos \delta_{13}}, \\ d_1 &= \frac{d_{12}^2}{u^2 + 1 - 2u \cos \delta_{12}}, \end{aligned}$$

which is equivalent to two quadratic equations in u and v , one we call E3 involving d_{23} and d_{13} and one we call E1, involving d_{13} and d_{12} . Solving E3 for u^2 and substituting in E1 allows E1 to be solved for u without involving radicals. Substituting u back in E3 yields a quartic in v , which can have as many as four real roots. For each v we obtain two roots for u through any of the quadratic equations, yielding a maximum of eight solutions [23.6, 7]. Popular pose estimation algorithms are based either on an iterative method [23.8, 9] or linear versions using auxiliary unknowns of higher dimension [23.10, 11].

23.1.2 Triangulation

When we know both the intrinsics and extrinsics or their summarization in the matrix \mathbf{P} and we measure a point we cannot recover its depth from just one camera position. Assuming that we have the projection of the same point \mathbf{X} in two cameras

$$\begin{aligned}\lambda_1 \mathbf{u}_1 &= \mathbf{P}_1 \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}, \\ \lambda_2 \mathbf{u}_2 &= \mathbf{P}_2 \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix},\end{aligned}\quad (23.7)$$

with known projection matrices \mathbf{P}_1 and \mathbf{P}_2 we can recover the position \mathbf{X} in space, a process well known as triangulation. Observe that we can achieve triangulation without decomposing the projection matrices into intrinsic and extrinsic parameters, although we need to remove the distortion in order to write them as above.

Having correspondences of the same point in two cameras with known projection matrices \mathbf{P}_1 and \mathbf{P}_r we can solve the two projection equations for the world point \mathbf{X} . It is worth noting that each point provides two independent equations so that triangulation becomes an overconstrained problem for two views. This is not a contradiction since two rays do not intersect in general in space unless they satisfy the epipolar constraint as presented in the next paragraph. The following matrix in the left-hand side has in general rank 4 unless the epipolar constraint is satisfied, in which case it has rank 3.

$$\begin{pmatrix} x\mathbf{P}_1(3, :) - \mathbf{P}_1(1, :) \\ y\mathbf{P}_1(3, :) - \mathbf{P}_1(2, :) \\ x\mathbf{P}_r(3, :) - \mathbf{P}_r(1, :) \\ y\mathbf{P}_r(3, :) - \mathbf{P}_r(2, :) \end{pmatrix} \begin{pmatrix} \mathbf{X} \\ Y \\ Z \\ 1 \end{pmatrix} = \mathbf{0}, \quad (23.8)$$

where $\mathbf{P}(i, :)$ means the i -th row of matrix \mathbf{P} .

Obviously, the homogeneous system above can be transformed into an inhomogeneous linear system with unknowns (X, Y, Z) . Otherwise it can be solved by finding the vector closest to the null-space of the 4×4 matrix above using singular value decomposition (SVD). A thorough treatment of triangulation is the classic [23.12].

23.1.3 Moving Stereo

Imagine now that a rigid stereo system consisting of cameras c_l (left) and c_r (right)

$$\mathbf{u}_{li} \sim \mathbf{P}_l \mathbf{X}_i, \quad (23.9)$$

$$\mathbf{u}_{ri} \sim \mathbf{P}_r \mathbf{X}_i, \quad (23.10)$$

is attached to a moving robot and observe this system at two time instances

$$\mathbf{X}_0 = \mathbf{R}_1 \mathbf{X}_1 + \mathbf{t}_1, \quad (23.11)$$

where \mathbf{X}_0 are point coordinates with respect to the world coordinate system, usually assumed to be with one of the camera instances, and \mathbf{X}_1 are the coordinates of the same point with respect to the camera rig, after a motion $(\mathbf{R}_1, \mathbf{t}_1)$. To estimate the motion of the rig, we have to solve two correspondence problems, first between the left and right image, and second between left (or right) at the first time instance and left (or right, respectively) at the second time instance. The left-to-right correspondence enables the solution of the triangulation problem at each time instance. Motion can then be obtained by solving (23.11) for $(\mathbf{R}_1, \mathbf{t}_1)$, a problem called absolute orientation. Alternatively one can avoid the second triangulation and solve the pose estimation problem between triangulated points in 3D and points in the left image only. In the robotics context, this is the setup most similar to the simultaneous localization and mapping (SLAM) (Chap. 37) problem when range sensors are used, and here we call it binocular SLAM.

Absolute Orientation

The treatment for moving stereo will be short and the reader is referred to a similar treatment in the chapter about range sensing. We assume that correspondences between two time instances have been established based on tracking in the images so that we can formulate equations of the form

$$\mathbf{X}_2 = \mathbf{R} \mathbf{X}_1 + \mathbf{t}.$$

The standard way [23.13, 14] to solve this problem is to eliminate the translation by subtracting the centroids, yielding

$$\mathbf{X}_2 - \bar{\mathbf{X}}_2 = \mathbf{R}(\mathbf{X}_1 - \bar{\mathbf{X}}_1).$$

We need at least three points in total to obtain at least two non-collinear mean-free $\mathbf{X} - \bar{\mathbf{X}}$ vectors. If we concatenate the mean free vectors for n points into an $3 \times n$ matrix $\mathbf{A}_{1,2}$ we can formulate the following minimization of the Frobenius norm

$$\min_{\mathbf{R} \in SO(3)} \|\mathbf{A}_2 - \mathbf{R} \mathbf{A}_1\|_F,$$

which is known as the Procrustes problem. It can be shown [23.14] that the solution is obtained through SVD

as

$$\mathbf{R} = \text{sign}(\det(\mathbf{U}\mathbf{V}^\top))\mathbf{U}\mathbf{V}^\top, \quad (23.12)$$

where \mathbf{U} , \mathbf{V} are obtained from the singular value decomposition

$$\mathbf{A}_2\mathbf{A}_1^\top = \mathbf{U}\mathbf{S}\mathbf{V}^\top.$$

Solutions are usually obtained with **RANSAC** by sampling triples of points and verification with the Procrustes method.

23.1.4 Structure from Motion (SfM)

Relax now the assumption that the projection matrices are known and focus on measuring and matching the corresponding points \mathbf{u}_1 and \mathbf{u}_2 . This is the well-known structure-from-motion problem or, more precisely, structure and 3-D motion from 2-D motion. In photogrammetry, this is well known as the relative orientation problem. Even after eliminating the λ 's from (23.9) or by writing them in projective equivalence form

$$\begin{aligned} \mathbf{u}_1 &\sim \mathbf{P}_1 \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}, \\ \mathbf{u}_2 &\sim \mathbf{P}_2 \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix}, \end{aligned} \quad (23.13)$$

we realize that, if $(X, Y, 1)$ is a solution, then $(\mathbf{H}X, \mathbf{P}_1\mathbf{H}^{-1}, \mathbf{P}_2\mathbf{H}^{-1})$ is also a solution, where \mathbf{H} is an invertible 4×4 real matrix or in other words a collineation in \mathbb{P}^3 . Even if we align the world coordinate system with the coordinate system of the first camera, which is common in practice,

$$\begin{aligned} \mathbf{u}_1 &\sim (1 \ 0) X, \\ \mathbf{u}_2 &\sim \mathbf{P}_2 X, \end{aligned} \quad (23.14)$$

we retain the same ambiguity, where \mathbf{H} is of the form

$$\mathbf{H} \sim \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ h_{41} & h_{42} & h_{43} & h_{44} \end{pmatrix}, \quad (23.15)$$

with $h_{44} \neq 0$. This ambiguity is possible when the projection matrices are arbitrary rank-3 real matrices without any constraint on their elements. If we assume that we have calibrated our cameras then the projection matrices depend only on displacements

$$\begin{aligned} \mathbf{u}_1 &\sim (1 \ 0) X, \\ \mathbf{u}_2 &\sim (\mathbf{R} \ t) X, \end{aligned} \quad (23.16)$$

and the only remaining ambiguity is the scale ambiguity, where \mathbf{H} looks like an identity matrix except with $h_{44} = s \neq 1$ being the scale factor. In other words, if (\mathbf{R}, t, X) is a solution, then $(\mathbf{R}, st, 1/sX)$ is a solution, too. These remarks generalize to multiple views. Because in robotics the (\mathbf{R}, t) matrices correspond to location and X to mapping of the environment, the problem is more properly described as simultaneous localization and mapping (**SLAM**). However, because the term **SLAM** has been used with a variety of sensors, such as sonar and laser range scanners, the term monocular **SLAM** is better suited to describe structure from motion based on multiple views [23.15].

Epipolar Geometry

This is probably one of the most studied problems in computer vision. We constrain ourselves to the calibrated case, which is most relevant to robotics applications. The necessary and sufficient condition for the two rays $\mathbf{R}\mathbf{x}_1$ and \mathbf{x}_2 to intersect is that the two rays are coplanar with the baseline t :

$$\mathbf{x}_2^\top (t \times \mathbf{R}\mathbf{x}_1) = 0, \quad (23.17)$$

which is the epipolar constraint (Fig. 23.2). To avoid the scale ambiguity we assume that t is a unit vector. We proceed by summarizing the unknowns into one matrix

$$\mathbf{E} = \hat{t} \mathbf{R}, \quad (23.18)$$

where \hat{t} is the 3×3 skew-symmetric matrix to the vector t . The \mathbf{E} matrix is called the essential matrix. The epipolar constraint then reads

$$\mathbf{x}_2^\top \mathbf{E} \mathbf{x}_1 = 0, \quad (23.19)$$

which is the equation of a line in the \mathbf{x}_2 plane with coefficients $\mathbf{E}\mathbf{x}_1$ or a coefficient of a line in the \mathbf{x}_1 plane with

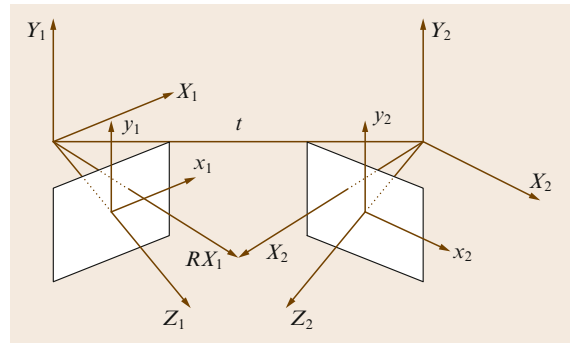


Fig. 23.2 Two views illustrating the coordinate transformations and the perspective projection of the world to a camera

coefficients $E^\top x_2$. These lines are called epipolar and form pencils whose centers are the epipoles e_1 and e_2 in the first and second image plane, respectively. Looking at Fig. 23.2, we can see that the epipoles are the intersections of the baseline with the two image planes, hence $e_2 \sim t$ and $e_1 \sim -R^\top t$. Looking at the equations of the epipolar lines we can immediately infer that $E^\top e_1 = 0$ and $Ee_2 = 0$.

The set of all essential matrices

$$\mathcal{E}_2 = \{E \in \mathbb{R}^{3 \times 3} \mid E = [t]_\times R, \text{ where } t \in S^2 \text{ and } R \in O(3)\} \quad (23.20)$$

has been characterized as a manifold of degree five. The following result has been proven [23.16].

Proposition 23.1

A matrix $E \in \mathbb{R}^{3 \times 3}$ is essential if and only if it has two singular values equal to each other and the third singular value equal zero.

We present here a very recent method introduced by Nister [23.17] for recovering an essential matrix from five point correspondences, which has gained in popularity because of its suitability for RANSAC methods.

Minimal Case

We expand the epipolar constraint in terms of the homogeneous coordinates $x_1 = (x_1, y_1, z_1)$ and $x_2 = (x_2, y_2, z_2)$ (when the points are not at infinity $z_i = 1$):

$$\begin{pmatrix} x_1 x_2^\top & y_1 x_2^\top & z_1 x_2^\top \end{pmatrix} E_s = 0, \quad (23.21)$$

where E_s is the row-by-row stacked version of matrix E .

When we use only five point correspondences the resulting linear homogeneous system will have as a solution any vector in the four-dimensional kernel of the data matrix:

$$E_s = \lambda_1 u_1 + \lambda_2 u_2 + \lambda_3 u_3 + \lambda_4 u_4. \quad (23.22)$$

At this point we want the matrix E resulting from E_s to be an essential matrix satisfying Proposition 23.1. It has been proven [23.16] that

Proposition 23.2

A matrix $E \in \mathbb{R}^{3 \times 3}$ is essential if and only if

$$EE^\top E = \frac{1}{2} \text{trace}(EE^\top) E. \quad (23.23)$$

Though the $\det(E) = 0$ constraint can be inferred from (23.23) we are still going to use it together with (23.23) to use ten cubic equations in E . As described in [23.17], one can obtain a tenth-degree polynomial in λ_4 . The number of real roots of this polynomial are computed with a Sturm sequence. There is no proof that a real root will exist at all, beyond the physical plausibility of the existence of at least one solution.

Assuming that we have recovered an essential matrix from point correspondences, the next task is to recover an orthogonal matrix R and a unit vector translation t from the essential matrix. It can be shown that, if $E = USV^\top$ is the singular value decomposition (SVD) of E with $\det(U) > 0$ and $\det(V) > 0$, then t is parallel to the last column of U and R is $UR_{z\pi}V^\top$ or $UR_{z\pi}^\top V^\top$, where $R_{z\pi}$ is a rotation of $\pi/2$ about the z -axis. Each of the rotations is equal to the other followed by a π -rotation around the baseline. The correct pairing of rotation and translation is chosen so that the reconstructed points are in front of the cameras.

Ambiguities

The approach with five point correspondences has a finite number of feasible (feasible means that they may produce multiple interpretations of structures in front of the camera) solutions when the points in the scene lie on a plane (a twofold ambiguity) [23.18] or when the points on the scene and the camera centers lie on a double sheet hyperboloid with the additional constraint that the camera centers lie symmetrically to the main generator of the hyperboloid [23.19]. These are inherent ambiguities which hold for any number of point correspondences when one seeks a solution for an exact essential matrix.

When solving the linear least-squares system for the essential matrix, a planar scene as well as the case of all points and the camera centers lying on a quadric causes a rank deficiency of the system and thus infinite solutions for E .

Beyond the ambiguous situations, there is a considerable amount of literature regarding instabilities in the two-view problem. In particular, it has been shown [23.18, 20, 21] that a small field of view and insufficient depth variation can cause an indeterminacy in the estimation of the angle between translation and optical axis. An additional small rotation can cause a confounding effect between translation and rotation [23.22]. Moreover, it has been shown that there exist local minima close to the global minimum that can fool any iterative scheme [23.23, 24].

23.1.5 Monocular SLAM or Multiple-View SfM

When we talk about simultaneous localization and mapping we obviously mean over a longer period of time. The question is how to integrate additional frames into our 3-D motion estimation (localization) process.

To exploit multiple frames we introduce rank constraints [23.25]. We assume that the world coordinate system coincides with the coordinate system of the first frame and that a scene point is projected onto \mathbf{x}_i in the i -th frame and that its depth with respect to the first frame is λ_1 :

$$\mathbf{x}_i = \mathbf{R}_i(\lambda_1 \mathbf{x}_1) + \mathbf{t}_i. \quad (23.24)$$

Taking the cross product with \mathbf{x}_i and writing it for n frames yields a homogeneous system

$$\begin{pmatrix} \hat{\mathbf{x}}_2 \times \mathbf{R}_2 \mathbf{x}_1 & \hat{\mathbf{x}}_2 \mathbf{t}_2 \\ \vdots & \vdots \\ \hat{\mathbf{x}}_n \times \mathbf{R}_n \mathbf{x}_1 & \hat{\mathbf{x}}_n \mathbf{t}_n \end{pmatrix} \begin{pmatrix} \lambda_1 \\ 1 \end{pmatrix} = 0 \quad (23.25)$$

that has the depth of a point in the first frame as an unknown. The $3n \times 2$ multiple view matrix has to have rank one [23.26], a constraint that infers both the epipolar and the trifocal equations. The least-squares solution for the depth can easily be derived as

$$\lambda_1 = - \frac{\sum_{i=1}^n (\mathbf{x}_i \times \mathbf{t}_i)^\top (\mathbf{x}_i \times \mathbf{R}_i \mathbf{x}_1)}{\|\mathbf{x}_1 \times \mathbf{R}_1 \mathbf{x}_1\|^2}. \quad (23.26)$$

Given a depth for each point we can solve for motion by rearranging the multiple-views constraint (23.25) as

$$\begin{pmatrix} \lambda_1^1 \mathbf{x}_1^{1\top} \otimes \hat{\mathbf{x}}_i^1 & \hat{\mathbf{x}}_i^1 \\ \vdots & \vdots \\ \lambda_1^n \mathbf{x}_1^{n\top} \otimes \hat{\mathbf{x}}_i^n & \hat{\mathbf{x}}_i^n \end{pmatrix} \begin{pmatrix} \mathbf{R}_i^{\text{stacked}} \\ \mathbf{t}_i \end{pmatrix} = 0 \quad (23.27)$$

where \mathbf{x}_i^n is the n -th image point in the i -th frame and $\mathbf{R}_i, \mathbf{t}_i$ is the motion from the first to the i -th frame and $\mathbf{R}_i^{\text{stacked}}$ is the 12×1 vector of stacked elements of the rotation matrix \mathbf{R}_i . Suppose that \mathbf{k} is the 12×1 kernel (or closest kernel in a least-squares sense) of the $3n \times 12$ matrix in the left hand side obtained through singular value decomposition and let us call \mathbf{A} the 3×3 matrix obtained from the first nine elements of \mathbf{k} and \mathbf{a} the vector of elements 10–12. To obtain a rotation matrix we follow the SVD steps in the solution of absolute orientation (23.12) to find the closest orthogonal matrix to an arbitrary invertible matrix.

On top of such an approach, a bundle adjustment [23.27] minimizes the sum of all deviations between image coordinates and the backprojections of the points to be reconstructed. For N points and M motions, the sum of squares of $2N(M+1)$ residuals has to be minimized with respect to all 3-D coordinates and all motions modulo a universal scale, yielding $3N+6M-1$ unknowns. Lourakis [23.28] exploits the sparse structure of the Jacobian involved in any nonlinear minimization. It is worth mentioning that bundle adjustment, though extremely slow, captures the correlation between motion estimates and structure (3-D points) estimates which is artificially hidden in the iterative scheme in (23.25).

The largest-scale motion estimation and registration of views have been performed by Teller [23.29] with a decoupled computation first of relative rotations and finally of relative translations. The above multiple-view SfM techniques can only be applied in a sliding-window mode in time due to their batch nature. Davison [23.15] showed the first real-time recursive approach by decoupling the direction of the viewing rays from the depth unknowns.

23.1.6 Dense Depth Maps from Stereo

In this section we treat a particular aspect of the ‘M’ in the visual SLAM, namely how we can obtain a *dense* map of the environment from two simultaneous (stereo) or consecutive images (structure from motion). We emphasized the density of the map to differentiate this task from visual SLAM approaches establishing only layout maps of landmark points.

We start with a simple explanation, namely, how to extract dense depth maps from a pair of cameras with parallel coordinate axes. Given the projection matrices or the essential matrix of two views we can always rotate the two cameras so that corresponding points lie on the same image row, a process called *rectification* [23.30]. Knowledge of the epipolar geometry is sufficient for rectification. We already described how to triangulate given a correspondence so the focus of this section is really solving the matching problem: for each pixel in the image what is the most similar point in the right image, and vice versa. Assume that we have a *similarity function* between two neighborhoods in the left I_l and right I_r images, respectively, whose central pixels differ by a disparity d . Each similarity measure defines a function of (x_l, d) called the *disparity space image* [23.31, 32]. Recent approaches based on the plane-sweeping method [23.33–35] use

a different domain for computing the correlation, which can be the backprojection of the image on frontoparallel planes and using the interdistance of these layers instead of the disparity. Such a backprojection requires the epipolar geometry and contains a different rectification. The choice of the *similarity* or *matching cost* function and the local *aggregation* directly affects the disparity space image and, depending on the time resources, can become quite nonlinear and elaborate. Of particular concern is the aggregation, which implicitly assumes that disparity is constant over a neighborhood. Such an assumption produces several artifacts at discontinuities unless offsetted bilateral aggregation is applied [23.36].

Given a disparity space image, algorithms differ from each other in the way they assign a disparity d to each point x_1 based on a local or global *optimization*. In local algorithms, the decision for x_1 is independent of that for other points in the image and the classical procedure is the greedy one, yielding the most similar pixel in the same scanline. Global algorithms solve either an entire image row (scanline) at once (dynamic programming) [23.31, 36, 39] or formulate a cost functional over the entire image that consists of the data term, a regularizing smoothness term, and a discontinuity-preserving step. Labeling all pixels with a disparity label is an *NP-hard* problem and the two dominant approximation paradigms are graph cuts [23.40, 41] or belief propagation [23.42–44].

The challenges in disparity computation arise from three circumstances: pixels might be occluded in one of the images (occlusions), image variation might be minimal (texturelessness), and the appearance of the same point/area might change due to perspective foreshortening or even due to violation of the Lambertian assumption. The latter effect is rather prominent in specularities and is accentuated when the baseline increases.

Occlusions are handled by local algorithms through bilateral windows and left–right consistency checks for the same correspondences independent of which of the two frames is used as reference. Occlusions are best processed by the dynamic programming approach (illustrated in Fig. 23.3), which we illustrate because of its real-time appeal. Applying dynamic programming in stereo presumes that disparity is a monotone function of the image positions, which means that, if we have x_{11}, d_1 and x_{12}, d_2 (two point-disparity pairs) and $x_{11} < x_{12}$ then $d_1 < d_2$. A further constraint subsumed by all algorithms is the uniqueness constraint, which dictates that a point in the left can have only one corresponding point in

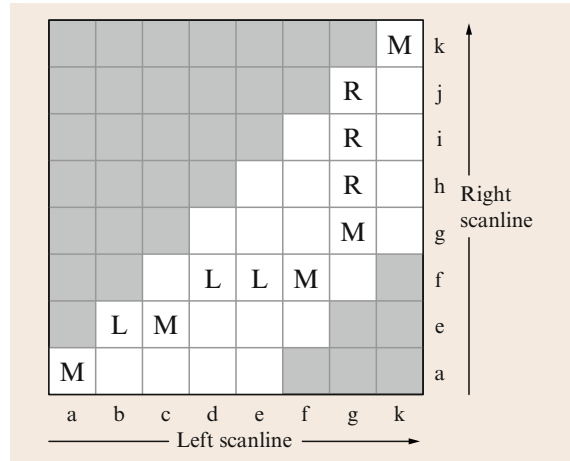


Fig. 23.3 The dynamic programming approach in stereo is based on finding the optimal path in the cost matrix of the picture. Matches are denoted by ‘M’, while ‘L’ and ‘R’ indicate visibility in the left or right image only, respectively [23.37]

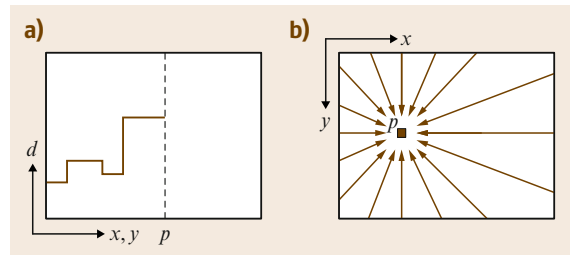


Fig. 23.4 (a) Minimum-cost path in disparity space. (b) Aggregation of costs in 16 directions [23.38]

the right, and vice versa. Several attempts have been made to overcome its single-scanline optimization character. One of the most successful and ranking highly in the stereo vision benchmark is the semiglobal approach in [23.38, 45] which aggregates the costs of scanlines in many directions (Fig. 23.4).

When multiple concatenated stereo pairs are used, for example, for a given a camera trajectory. We have to combine all depth maps into one 3-D model that is as consistent as possible with all the views. Multiple-view stereo [23.46] takes advantage of the existence of multiple cameras around an object by computing the visual hull of objects and refining depth by applying photoconsistency [23.47]. In the context of robotics, we rarely have such a surround capture but we definitely have multiple views captured by a moving system with estimated motion.

23.2 Recognition

Whatever task a robot performs it must be able to determine where things are in the environment and also identify relevant objects, structures, and events. Vision provides rich information about both *where* and *what*, and visual recognition is therefore an essential capability for robots. However, generic visual object recognition is far from easy. It is not even easy to define, since what constitutes an object in the world is not trivial to define without additional constraints (see, e.g., [23.48]). The problem becomes even more difficult if one talks about classes or categories of objects or, say, places. Visual recognition lies at the heart of computer vision research and a vast number of methods have been proposed to deal with it. Still, much remains unknown about how to realize it.

Fortunately, in many cases in robotics we are not faced with the most general aspects of object recognition or classification. More precisely, the problem is often to find an object that either is known or was recently seen. This is typically the case in tracking, navigation, and manipulation. Other situations concern recognizing landmarks or structures such as roads, or more generally places. These tasks all involve recognition, but mainly in the sense of establishing correspondence between phenomena in different images or between an image and a model. Other cases concern the determination of whether something belongs to a class of objects or establishing whether what is seen is a particular type of object, e.g., a road. This is usually called object classification or categorization and constitutes an even more difficult problem. The most general form of the categorization problem, of course, also includes forming the categories, but that is beyond this treatment.

The literature on computer-based visual object recognition, classification, and categorization is very rich and a multitude of methods have been proposed since the very early days of computer vision. Characteristically certain approaches have waxed and waned in popularity. For instance, the purely statistical pattern recognition methods advocated during the early days, but discounted around 1970–80 when 3-D reconstruction and physical and geometrical modeling were the focus, are again being strongly emphasized, not least due to advances in machine learning and increasingly powerful computers. Surveying all existing techniques would require volumes rather than chapters. In the next few sections we will therefore describe some of the methods that seem particularly useful in the type of

robot applications that we have mentioned. The focus on these applications has resulted in many interesting approaches being left out. A comprehensive set of references has been given as entry points to several of these techniques and to the extensive literature on the subject. Moreover, good surveys can be found in the slides from the short course by *Fei Fei et al.* [23.49] or in *Pinz* [23.50], even though they treat categorization rather than recognition.

23.2.1 Approaches to Recognition

Early approaches to object recognition considered objects as represented by 3-D models or by a decomposition into surfaces or volumetric primitives. The first attempts by *Roberts* and *Guzman* [23.51] considered simple geometric parts. *Binford* [23.52] introduced generalized cylinders, which were used by *Brooks* [23.53] in the ACRONYM system and later in the model of *Marr* and *Nishihara* [23.54]. These approaches were based on object-centered representations to enable view invariance. *Marr* [23.55] and others assumed that these could be obtained by 3-D reconstruction from 2-D images using stereo or monocular cues, or by direct acquisition of range images. Several such systems were presented, e.g., *Faugeras et al.* [23.56] and *Bolles et al.* [23.57]. *Biederman* [23.58] introduced his recognition-by-components (RBC) theory as a model for human object recognition from 2-D images. This work inspired numerous attempts to implement RBC-based systems on computers. However, extracting the needed geometric primitives from images turned out to be difficult. The first step of edge detection seldom led to robust indications of where to find the parts. In the ACRONYM system this was addressed by advanced reasoning, in other systems, such as those of *Mohan and Nevatia* [23.59] and *Zisserman et al.* [23.60], edges were grouped to generate part hypotheses. Another system using the same principle, but going for the object without any intermediate part representation was, proposed by *Nelson and Selinger* [23.61]. Increasingly advanced methods for representing and analyzing shapes based on skeletons have also been proposed. These usually require a silhouette curve and therefore also assume prior segmentation. The influence of these techniques on object recognition has consequently been limited at least insofar that they assume object-centered representations.

The general development has instead been towards viewer-centered representations. In fact, there is an on-

going debate on the role of such representations in human recognition (see, e.g., *Tarr and Bülthoff* [23.62]). In computer vision, methods of this sort go back to very early pattern-recognition-based approaches. A revived interest in these first arose in work on face recognition, e.g., *Turk and Pentland* [23.63]. This was paralleled by advances on neural network models, e.g., *Poggio and Edelman* [23.64]. By showing that training a system on different views of a set of objects resulted in excellent recognition rates *Murase and Nayar* [23.65] initiated an interest in learning and statistical approaches that has continued and developed substantially ever since. Notably the focus has changed from global to local methods, which was already suggested by *Rao and Ballard* [23.66]. Discriminative as well as generative models have been proposed, which in turn has implied that the methods apply to object categorization as well as recognition. Moreover, more explicit representation of structure and parts have also been incorporated. This introduction of what is now often called constellation-based approaches has in fact decreased the difference between part- and appearance-based methods and again implied a return to very early methods such as the spring-loaded template matching proposed by *Fischler and Elschlager* [23.67]. Nevertheless, we will in the sequel describe methods as being appearance-based or constellation- or part-based as a way of stressing different aspects of them.

23.2.2 Appearance-Based Methods

Appearance-based approaches to object recognition rely on the extraction of distinctive features in the images. In pattern recognition and image processing features are defined as N -tuples or vectors whose components are functions of the initial pattern variables or a subset of them (from *Haralick and Shapiro* [23.68]). Features can be computed both locally and globally in the images, but in robotics applications local features tend to be most appropriate. In matching one needs to find distinguishing features that characterize the patterns one is looking for. Hence, one separates the steps of detecting features and computing feature descriptors.

As described in the section on *Ambiguities* features computed at discrete locations of the images should preferably be invariant to observer and scene motion. Therefore, geometric features, such as points, lines, and curves have been extensively used, both in computer vision and robotics. These types of features can be computed using derivatives of the image function. Also more general derivative-based features can be derived (see

Lindeberg [23.69]), and in fact sets of differential invariants that completely represent the image function can be defined. However, simple point features and even extended features such as lines only partly represent the visual information at their locations. Therefore, richer descriptors of these locations are often needed to address, e.g., the correspondence problem. Furthermore, invariance, or at least covariance, is desirable not only for motion or small viewpoint changes, but also for illumination variations and large-scale motions and distance changes. These issues have been considered in recent approaches to extraction of features and feature descriptors (*Lowe* [23.70], *Schmid and Mohr* [23.71], *Mikolajczyk and Schmid* [23.72], *Matas et al.* [23.73], and *Mikolajczyk et al.* [23.74]). Building on decades of development of feature extractors this work today provides a set techniques with rather well-studied performance. The features as well as the feature descriptors are generally computed over small regions around some geometric feature, say a point. In that sense region properties are accounted for, for instance, the scale-invariant feature transformation (*SIFT*) descriptor by *Lowe* captures local luminance variations. However, to achieve invariance to illumination the actual luminance values are eliminated in these approaches, and color is not used at all. By including color features additional properties are captured, even though computational models for color constancy, i.e., deriving surface color independently of illumination, have turned out to be very difficult. Color features are still valuable for discrimination. In general, the different features used in the literature are complementary in applicability both with regard to imaging conditions and scene content.

Distinctive Features

The idea of finding interest points for matching was pioneered by *Moravec* [23.75]. His method was furthered in the frequently used detector proposed by *Harris and Stephens* [23.76], usually called the Harris detector. It is based on the windowed second moment matrix,

$$\mathbf{M} = \mathbf{E} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} = \mathbf{E}((\nabla I)(\nabla I)^T). \quad (23.28)$$

The eigenvalues of the matrix in (23.28) are proportional to the principal curvatures of the autocorrelation function of I . When both these curvatures are high one has a corner in the image, that is moreover stable with respect to lighting variations. Generally, this detector represents the local distribution of orientations. It was independently introduced as such by *Förstner* [23.77]

and Bigün and Granlund [23.78]. Typically a Gaussian function is used both for windowing and differentiation, and then the operator takes the form

$$\begin{aligned} \mathbf{M} &= \mathbf{E}(\cdot, \sigma_I \sigma_D) \\ &= \sigma_D^2 G(\sigma_I) \begin{pmatrix} I_x^2(\cdot, \sigma_D) & I_x I_y(\cdot, \sigma_D) \\ I_x I_y(\cdot, \sigma_D) & I_y^2(\cdot, \sigma_D) \end{pmatrix} \end{aligned} \quad (23.29)$$

where σ_I and σ_D are the scales of the Gaussian kernels for windowing and differentiation. This matrix is positive semidefinite with eigenvalues $\lambda_1, \lambda_2 \geq 0$. Its trace

$$\mathbf{M} = \lambda_1 + \lambda_2 \quad (23.30)$$

represents the strength of the operator response. The direction of the eigenvector corresponding to the largest eigenvalue gives the average gradient direction in the neighborhood. Moreover

$$\det(\mathbf{M}) = \lambda_1 \lambda_2 \quad (23.31)$$

provides a measure of the spread of directions around the point. An alternative to the Harris detector is given by the Hessian matrix

$$\mathbf{H} = \mathbf{H}(\cdot, \sigma_I \sigma_D) = \begin{pmatrix} I_{xx}(\cdot, \sigma_D) & I_{xy}(\cdot, \sigma_D) \\ I_{xy}(\cdot, \sigma_D) & I_{yy}(\cdot, \sigma_D) \end{pmatrix}. \quad (23.32)$$

This operator gives somewhat stronger responses at blobs and ridges (see, e.g., Lindeberg [23.69] or Lowe [23.79] for details), but of course requires second-order derivatives. The eigenvalues are in this case proportional to the principal curvatures of the greylevel function and can again be computed from the trace and the determinant. Lowe [23.70] uses this for a test on the ratio of the eigenvalues to reject keypoints that are on edges. We refer to his paper for details.

Robust Methods for Finding Distinctive Points

The response of these operators depend on and are sensitive to scale changes. To find distinctive points suitable for matching one therefore needs to find the points of maximum response in both space and scale, i.e., search for maxima in a 3-D space. Lindeberg [23.80] presents a method for this based on the notion of characteristic scale. Mikolajczyk and Schmid [23.72] and Lowe [23.70, 79] later described efficient ways of implementing this. Mikolajczyk and Schmid first use the Harris detector to localize points in 2D and then select the points at which the Laplacian attains a maximum over scales as the final set. This detector, the Harris–Laplacian, is shown to give the best repeatability over

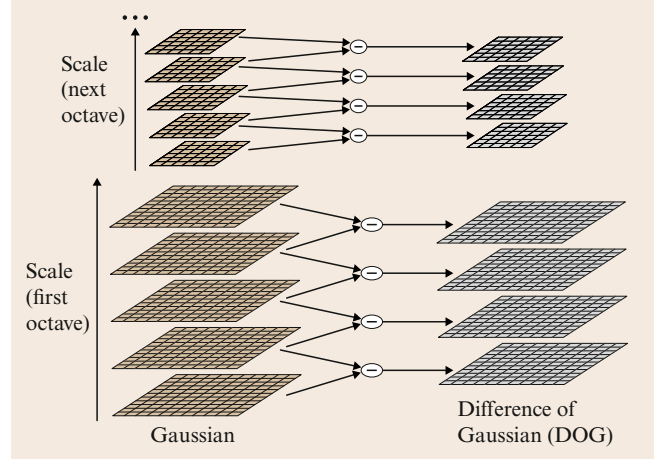


Fig. 23.5 For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian (DOG) images on the right. After each octave, the Gaussian image is downsampled by a factor of 2, and the process is repeated

scale changes among several other common interest point detectors.

Lindeberg showed that one needs to normalize the Laplacian with the factor σ to obtain scale invariance, and indeed Mikolajczyk [23.72] demonstrated experimentally that the extrema of $\sigma^2 \nabla^2 G$ produce the over scale most stable image features of several operators. Lowe [23.70] presents an efficient approximate method for computing these extrema using D , the difference of Gaussians (DOG) instead. This scale space computation is shown in Fig. 23.5. Introducing the factor k he writes

$$D(\cdot, \delta) = G(\cdot, kt) - G(\cdot, t). \quad (23.33)$$

From the heat diffusion equation it follows that

$$\frac{\partial G}{\partial t} = t \nabla^2 G. \quad (23.34)$$

Hence

$$\partial \nabla^2 G = \frac{\partial G}{\partial t} \approx \frac{D}{kt - t} \quad (23.35)$$

that is

$$D \approx (k - 1) \delta^2 \nabla^2 G. \quad (23.36)$$

Lowe's efficient method is based on using the approximation to find the points using a set of DOG images. The result of these operations is a set of keypoints with associated location (x - and y -coordinates),

scale, and orientation. The described operators are not affinely invariant, but can be made so, see e.g. *Mikolajczyk et al.* [23.74]. However, as pointed out by Lowe one can adapt the descriptor so that this becomes unnecessary in many cases, at least if the scene contains enough distinctive features.

Distinctive Regions

Matas et al. [23.73] proposed a method for finding distinguished regions that has turned out to give good results on wide-baseline matching and also in applications to object recognition. It is based on maximally stable extremal regions (**MSER**), that is connected components of thresholded images. The appropriate way to threshold will become clear below. The regions are extremal in the sense that all pixels inside the **MSER** have either higher (bright extremal regions) or lower intensity (dark extremal regions) than all the pixels on its boundary. They are stable in the sense that this property is optimized at threshold selection.

The set of extremal regions have a number of desirable properties, see *Matas et al.* [23.73]. For instance, they are unaffected by monotonic or affine intensity changes and to continuous geometric image transformations that preserve topology. Hence, they are stable with respect to many geometric and photometric changes that are common and not always known well enough to be corrected.

To describe the algorithm we follow *Matas et al.* [23.73]. On a digital image I defined on \mathbb{D} with values in \mathbb{S} , typically 0–255 regions are defined as connected components under the topology given by 4-neighborhoods. An extremal region \mathbb{D} is a region such that $p \in \mathbb{D}, q \in \partial\mathbb{D} : I(p) > I(q)$ (maximum intensity region) or $I(p) < I(q)$ (minimum intensity region).

To define a maximally stable extremal region we consider a nested set of regions $D_1 \subset \dots \subset D_i \subset D_{i+1} \subset \dots$. Region D_i^* is maximally stable if q has a maximum at i , where

$$q(i) = |D_{i+h}| \setminus |D_{i-h}| / |D_i| \quad (23.37)$$

and $|\cdot|$ denotes cardinality; h is a parameter of the method.

The nested set of regions is obtained by successively thresholding I with a threshold that transcends \mathbb{S} , e.g., steps from 0 to 255. The regions obtained from such an operation can be enumerated in a straightforward manner. First the pixels are sorted by intensity. Then the pixels are placed in the image, either in decreasing or increasing order, and the list of connected components

and their areas are maintained using, e.g., an efficient union-find algorithm [23.81]. If \mathbb{S} is a small discrete set, such as 0, ..., 255 this algorithm is almost linear in the number of pixels and allows an efficient implementation. Notably, the extremal regions can be messy, but the **MSER** operation only retains rather simple regions, which is important for subsequent computation of feature descriptors.

The described method finds salient structure without using derivatives. Other techniques of this nature also exist, e.g., the detector by *Kadir and Brady* [23.82], the SUSAN corner detection by *Smith and Brady* [23.83] and morphological methods.

Descriptors for Distinctive Points and Regions

As mentioned above interest point as well as region detectors provide information about local regions over which suitable descriptors can be computed. Hence, Lowe uses scale information to determine an elliptical region, *Matas et al.* the convex hull of the **MSER**, and *Mikolajczyk et al.* perform an affine normalization to obtain comparable regions in different images.

Numerous region-based descriptors have been proposed for appearance-based object recognition. These have included methods using the local image patch itself [23.84], its statistics (*Schiele and Crowley* [23.85]), and more generally filtered versions of it (*Rao and Ballard* [23.66]). The success of the second-order moment matrix in motion and shape computation indicates the importance of utilizing the directional statistics. *Lowe* [23.70] suggested a particularly useful descriptor in his scale-invariant feature transformation (**SIFT**) features.

These descriptors are computed at the distinctive points found, e.g., by the methods described earlier in this section, which determine keypoints with an associated scale. To compute them an orientation is first assigned to the keypoint so as to achieve invariance to image rotation. Notably, this is different from using rotationally invariant measures, as in *Schmid and Mohr* [23.71]. *Lowe* [23.70] computes the orientation by finding peaks in the histograms of gradient orientations, with the entries weighted by gradient magnitude. The peaks correspond to dominant directions of local gradients. More than one peak can be accepted if several exist that almost reach the highest value. As a consequence several feature descriptors can be assigned to a single image point. Lowe sets the threshold to 80% of the maximum and keeps up to two peaks. Experiments show that as many as four peaks can be useful.

We refer readers to [23.70] for more details on the computations.

Given these steps we have a set of keypoints with associated locations, scales, and orientations. At each point in a window around such a keypoint we again compute the gradient magnitude and orientation. Weighting the values with a Gaussian we histogram them in $n \times n$ subregions in a $k \times k$ pattern around the keypoint; see Fig. 23.6. Lowe [23.70] in his experiments uses $n = k = 4$. In the figure $n = 4$ and $k = 2$. The number of bins for orientation is eight. More details on the computations and the choice of parameters are given in the paper. An important aspect is the choice of measurement region and how it is related to the scale at which the keypoint is detected. This is discussed in the paper, but also in Matas et al. [23.73] and in Mikolajczyk et al. [23.74]. In the latter paper affine normalization is introduced in the comparison to affine invariant detectors. Lowe discusses the usefulness of that in his article as well.

The notion of considering directional statistics around keypoints has also been used in character recognition, where of course the idea of representing the objects (the characters) as patterns arranged around a central point is natural.

Patches and Local Histograms as Features

An alternative and a complement to using features of the kind described above is to use the image patches directly as local descriptors. Such methods can of course be preceded by a step for determining interest points to limit the amount of computation. Agarwal and Roth [23.86] proposed a method based on a codebook of patches that later was furthered by Leibe et al. [23.84] and several other authors.

The patch preserves local image structure, but it has been shown that also histograms of the information in the patch might be sufficient for object recognition and localization. Swain and Ballard [23.87] showed examples of recognition based on comparisons of color information. However, they used red–green–blue (RGB) data directly, which turns out to be sensitive to illumination variations. Schiele and Crowley [23.85] instead used histograms of the output of receptive field computations, either first-order Gaussian derivative operators or differential invariants at three scales. Schneiderman and Kanade [23.88] showed that efficient recognition of faces and cars could be obtained from histograms of wavelet coefficients. More recently Linde and Lindeberg [23.89] introduced higher-order histograms and showed that they can be efficiently computed. Gener-

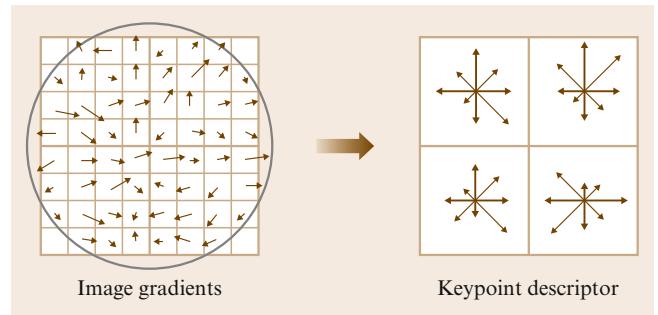


Fig. 23.6 A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. The samples are then accumulated into orientation histograms summarizing the contents over 4×4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. This figure shows a 2×2 descriptor array computed from an 8×8 set of samples, whereas the experiments in this chapter use 4×4 descriptors computed from a 16×16 sample array

ally, these types of methods are simple to use and give good performance provided that sufficient local variations are at hand. A more general framework for such methods has been proposed by Koenderink and van Doorn [23.90], introducing the notion of locally orderless images. Methods for comparing histograms are well known in statistics. An excellent and concise overview of such techniques is given in Rubner and Tomasi [23.91].

23.2.3 Matching

Visual recognition implies matching features or quantities derived from an image to stored representations of objects or images. This is a classical and extensively studied problem that is treated in most standard textbooks on image and signal processing as well as pattern recognition. With new feature-based techniques and with the applications to increasingly large data sets it has attracted considerable interest in recent years and novel contributions have appeared. Some of these will be discussed in the section on constellation based methods. Here we consider some approaches to matching in feature- and appearance-based recognition.

Vocabulary-Based Methods

Straightforward matching using descriptors such as SIFT implies finding nearest neighbors in high-dimensional spaces, which is computationally difficult.

Therefore, Lowe and others use clustering methods to create what can be regarded as an alphabet. Information represented in this way is amenable to general information retrieval methods.

To index an image database *Sivic* and *Zisserman* [23.92] introduced the idea to vector-quantize region descriptors into clusters using k -means and letting these clusters serve as visual words in a text retrieval approach. Given a vocabulary generated from a set of training images descriptors are extracted from each new image and assigned to the nearest cluster. In this way matches are immediately obtained for each new image. *Sivic* and *Zisserman* applied this to video retrieval, but the method is useful also for other sets of images. Text retrieval is performed using term-frequency inverse document frequency (TF-IDF) relevance scoring (*Baeza-Yates* and *Ribiero-Neto* [23.93]). This technique has been generalized by *Nister* and *Stewenius* [23.94] in the method described next.

Recognition Using Vocabulary Trees

Nister and *Stewenius* [23.94] introduced a hierarchical TF-IDF. In this approach a vocabulary tree is defined by hierarchically defined words. They obtain a very efficient lookup of visual words and can therefore use a larger vocabulary that in turn is shown to give improved retrieval quality (Fig. 23.7).

The vocabulary tree is built up by hierarchical k -means clustering, where k defines a branch factor for the tree. A vector forming a branch of the tree represents a visual word. To compute the score of a new image one needs to determine how similar its descriptors are to the paths down the vocabulary tree. *Nister* and *Stewenius*

propose assigning a weight w_i to each node i in the tree, for example, based on entropy, and then define the query q and the database vector according to the assigned weights

$$q_i = n_i w_i , \\ d_i = m_i w_i ,$$

where n_i and w_i are the number of descriptors in the two images. The relevance score is then set to

$$S(q, d) = \left\| \frac{q}{\|q\|} - \frac{d}{\|d\|} \right\| . \quad (23.38)$$

Nister and *Stewenius* recommend the L_1 -norm, but any norm can be used. Furthermore, they suggest computing the weights as

$$w_i = \ln \frac{N}{N_i} ,$$

where N is the number of images in the database, and N_i is the number of images in the database with at least one descriptor vector path through node. The proposed method has been shown to give real-time high-quality performance for retrieval from databases of thousands of images.

A related method using vocabulary-based pyramid matching is presented in *Grauman* and *Darrell* [23.95]. They compute an approximate partial matching between two sets of feature vectors. It should be added that these methods mainly address the indexing problem and give the set of most similar images. For localization an additional step for more precise matching and alignment may be required.

High-Dimensional Feature Matching

As mentioned above feature matching in high dimensions is computationally difficult. However, useful progress on this problem has also been made. Theoretically it is known that exact identification of the nearest neighbor in high-dimensional spaces in general leads to exhaustive search. Approximative methods have been proposed, e.g., by *Beis* and *Lowe* [23.96] and *Indyk* and *Motwani* [23.97], but they need to trade off speed of computation for quality of approximation. A promising approach to deal with this problem has recently been proposed by *Omercevic* et al. [23.98]. Their method is based on the notion of meaningful nearest neighbors. Such neighbors should be sufficiently close to a query feature that it is an outlier to the background feature distribution. This idea is based on the finding that the tail of the distribution of random outliers can be modeled by

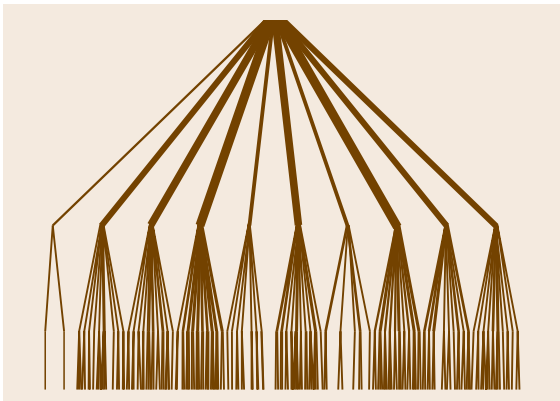


Fig. 23.7 Three levels of a vocabulary tree with branch factor 10 populated to represent an image with 400 features

an exponential and that nearest neighbors can therefore be weighted by how much they are similar to the query and dissimilar to the background. Dot products are used to measure the similarity between feature vectors. The authors also introduce a search method based on sparse coding and obtain an approximate method that outperforms other techniques either with respect to speed or accuracy. For instance, while slower than the vocabulary tree method it gave better recognition performance on the same data sets.

23.2.4 Constellation-Based Methods – Recognition by Parts

As mentioned above the success of feature and appearance-based methods together with the difficulty in defining and localizing parts have decreased interest in part-based techniques. However, the obvious advantage of using structural information has led to a renewed focus on the problem and in this subsection we will discuss some recent advances in this area. We will consider methods that use structural relations between the parts, but do not assume that the parts necessarily correspond to geometric primitives of some sort or even geometric parts at all. In fact, it turns out that in this way the distinction between part- and appearance-based becomes rather fuzzy, at least in cases when appearance refers to local parts of the object and not the entire object. Forsyth and Ponce in their book unify such methods by talking about them as being based on structural relations between templates, where the templates can be geometric and defined by 3-D or 2-D shape, or by visual appearance, e.g., given by intensity, color, or texture.

A structural model of an object can be given by a collection of parts together with a representation of how these are connected. Connections between pairs of parts are expressed by an undirected graph $G(V, E)$, where the vertices $v = \{v_1, \dots, v_n\}$ correspond to the n parts and the edge $(v_i, v_j) \in E$ implies that parts v_i and v_j are connected. An instance of an object is given by a configuration $L = (l_1, \dots, l_n)$, l_i being the location of part v_i . There are many ways of parameterizing the locations, including simple image positions or, e.g., positions and joining angles in 2D or 3D for the parts. The idea of using parts in this way is appealing in many ways. Parts provide a modular representation and they may be shared by many objects. Since parts can be simple they are less variable than the entire objects and also often less sensitive to pose variations. Furthermore, occlusion, clutter, and lighting variations

are unlikely to influence the recognition of all the parts.

However, it is important to cope with two problems. One has been discussed already, namely that of detecting and localizing the parts. The second one concerns matching, which is generally computationally difficult. *Fischler and Elschlager* [23.67] proposed to address it as an energy-minimization problem in the image domain. Such an approach is described by *Felzenszwalb and Huttenlocher* [23.99] as follows.

The cost or energy of a given configuration depends on how well a part matches the image data and on how well the parts agree with the model. Given an image, let $m_i(l_i)$ be a measure of the degree of mismatch when part v_i is placed at location l_i in the image. For a pair of connected parts let $d_{ij}(l_i, l_j)$ measure the degree of deformation of the model when part v_i has location l_i and part v_j location l_j . An optimal match of the model to the image can then be defined as

$$L^* = \arg \min_L \left(\sum_{i=1}^n m_i(l_i) + \sum_{(v_i, v_j) \in E} d_{ij}(l_i, l_j) \right). \quad (23.39)$$

This expression gives a configuration that minimizes the sum of match costs for each part and the deformation costs d_{ij} for connected pairs of parts. If the deformation costs only depend on the relative positions of the parts in the pairs the model is invariant to common global transformations.

A problem with this type of formulation is that it generally lacks efficient solutions. If the objects are complicated and especially if the amount of non-rigidity is high the number of parameters required is often also high. Modern approaches to these issues use probabilistic models. Hence, they rely on probabilistic estimates of where parts are rather than on deterministic segmentation steps. The estimates are obtained using learned representations of the appearance of the parts and priors on the configurations. This allows statistical formulations that include both the part finding and configuration problem. However, to initiate such algorithms one still has to find good candidate hypotheses with methods of the type described in the next section.

Let Θ be a set of parameters defining the object and L be the configuration, i.e., the location of each part. Then $P(I | L, \Theta)$ is the distribution that measures the likelihood of image I given a viewed object. From Bayes' rule follows that the posterior distribution of the

object configuration L given the model Θ and the image I is

$$P(L | I, \Theta) \propto P(I | L, \Theta)P(L | \Theta), \quad (23.40)$$

where $P(L | \Theta)$ is the prior probability that an object is at a particular location.

Following Felzenschwalb and Huttenlocher [23.99], the matching problem formulated as an energy-minimization problem can in this framework be regarded as an MAP estimation problem. They propose an efficient algorithm for this. In their framework the model parameters are $\Theta = (u, E, c)$, where $u = \{u_1, \dots, u_r\}$ are appearance parameters, E is the set of edges telling what parts are connected (in a graph representation), and $c = \{c_{ij} | (v_i, v_j) \in E\}$ are connection parameters. The model parameters are learned from a set of training images using maximum-likelihood estimation. Energy minimization in this formulation is generally NP-hard, but Felzenschwalb and Huttenlocher use the fact that the graph describing the structure has a restricted form to present an efficient algorithm. In other work Fergus et al. [23.100] parameterize with respect to appearance, location, and scale, not including any explicit graph representation of the configuration. Their method is aimed at recognizing classes of objects rather than single exemplars.

23.2.5 Place Recognition and Terrain Classification

Localization is a fundamental problem in mobile robotics. Two central aspects of this problem concern continuous pose maintenance and global localization, sometimes called the *robot kidnapping* problem. These problems are generally treated in the context of SLAM. Vision-based techniques form but one class of approaches and the way they are applied largely depends on the type of additional information that is available. In

any case, landmark detection and recognition as well as global localization can be addressed using vision and in both cases the problems then share many aspects with object recognition and image retrieval. Hence, several of the techniques presented in the previous sections also apply in these cases.

Scale-invariant key points and SIFT features have been used by Kosecka et al. [23.101]. They perform global localization indoors by recognizing locations and exploit information from neighborhood relations from a map using HMMs. Wolf et al. [23.102] propose a similar approach. Others have used histogram descriptors to represent the scenes. Ulrich and Nourbakhsh [23.103] compute color histograms from omnidirectional camera images and match them to stored images in combination with predictions from a topological map. In that way near-real-time performance is obtained through a simple voting process over the color bands. The method is successfully applied to indoor as well as outdoor environments. Davidson and Murray [23.104] used actively controlled cameras to find landmarks indoors following Bajcsy's active perception paradigm [23.105].

Learning is a central problem in place recognition. In most cases many training images are needed to obtain robust recognition. Moreover, additional information in terms of maps or approximate position is used to facilitate the recognition stage. Ramos et al. [23.106] propose a Bayesian approach to deal with these problems. Hence, their method can learn from few (3–10) training images and requires no map. Images are divided into patches and the world is interpreted as a set of places, each having a probabilistic representation. Matching is performed in near real time. They first perform dimensionality reduction on a patch representation of the scenes and then derive a generative probabilistic model of the output in terms of a set of linear mixture models through expectation maximization. The result of this inference process is a mixture of Gaussians that is used in a multiclass classification scheme in which the log likelihood for the model that best explains the given set of patches is selected.

23.3 Conclusion and Further Reading

As main additional sources of reading, we recommend the books by Hartley and Zisserman [23.5], Ma et al. [23.26], Faugeras [23.107], and Faugeras and Luong [23.4]. There is no textbook updated with the most recent results in recognition but the reader is encouraged

to browse through the ICCV'05 and CVPR'07 tutorials by Fei-Fei Li, Antonio Torralba, and Rob Fergus. The most representative and real-time implemented system to recognize instances of objects is Nister's [23.94] vocabulary tree approach.

Fig. 23.8 Reconstruction from a camera mounted on a vehicle (after 23.108) obtained from the fusion of 13 depth maps. Each depth map was obtained from 11-view stereo ►

We will close with a system of motion and dense mapping representing the state of the art: the work in [23.108] represents the state of the art in large-scale dense reconstruction from monocular image sequences without the use of an additional sensor. The camera poses are obtained by applying the algorithm described above [23.17]. A temporary model is obtained and subsequent poses are estimated by applying pose estimation with preemptive RANSAC [23.109] for a time window after which a reinitialization with a novel triple of views is obtained. A depth map is obtained by back-projecting multiple views on planes that sweep space in prominent directions and maximizing a correlation in the sweeping direction. Multiple depth maps are combined (Fig. 23.8) through median fusion and taking visibility into consideration [23.110].



References

- 23.1 Z. Zhang: A flexible new technique for camera calibration, *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 1330–1334 (2000)
- 23.2 M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, R. Koch: Visual modeling with a hand-held camera, *Int. J. Comput. Vis.* **59**, 207–232 (2004)
- 23.3 M. Pollefeys, L. Van Gool: Stratified self-calibration with the modulus constraint, *IEEE Trans. Pattern Anal. Mach. Intell.* **21**, 707–724 (1999)
- 23.4 O. Faugeras, Q.-T. Luong, T. Papadopoulos: *The Geometry of Multiple Images* (MIT Press, Cambridge 2001)
- 23.5 R. Hartley, A. Zisserman: *Multiple View Geometry* (Cambridge Univ. Press, Cambridge 2000)
- 23.6 K. Ottenberg, R.M. Haralick, C.-N. Lee, M. Nolle: Review and analysis of solutions of the three-point perspective problem, *Int. J. Comput. Vis.* **13**, 331–356 (1994)
- 23.7 M.A. Fischler, R.C. Bolles: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM* **24**, 381–395 (1981)
- 23.8 R. Kumar, A.R. Hanson: Robust methods for estimating pose and a sensitivity analysis, *Comput. Vis. Image Underst.* **60**, 313–342 (1994)
- 23.9 C.-P. Lu, G. Hager, E. Mjølness: Fast and globally convergent pose estimation from video images, *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 610–622 (2000)
- 23.10 L. Quan, Z. Lan: Linear n-point camera pose determination, *IEEE Trans. Pattern Anal. Mach. Intell.* **21**, 774–780 (1999)
- 23.11 A. Ansar, K. Daniilidis: Linear pose estimation from points and lines, *IEEE Trans. Pattern Anal. Mach. Intell.* **25**, 578–589 (2003)
- 23.12 R.I. Hartley, P. Sturm: *Triangulation. Computer Vision and Image Understanding* (1997)
- 23.13 B.K.P. Horn, H.M. Hilden, S. Negahdaripour: Closed-form solution of absolute orientation using orthonormal matrices, *J. Opt. Soc. Am. A* **A5**, 1127–1135 (1988)
- 23.14 G.H. Golub, C.F. van Loan: *Matrix Computations* (The Johns Hopkins Univ. Press, Baltimore 1983)
- 23.15 A.J. Davison, I.D. Reid, N.D. Molton, O. Stasse: Monoslam: Real-time single camera slam, *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(6), 1052–1067 (2007)
- 23.16 T.S. Huang, O.D. Faugeras: Some properties of the e matrix in two-view motion estimation, *IEEE Trans. Pattern Anal. Mach. Intell.* **11**, 1310–1312 (1989)
- 23.17 D. Nister: An efficient solution for the five-point relative pose problem, *IEEE Trans. Pattern Anal. Mach. Intell.* **26**, 756–777 (2004)
- 23.18 S. Maybank: *Theory of Reconstruction from Image Motion* (Springer, Berlin, Heidelberg 1993)
- 23.19 S.J. Maybank: The projective geometry of ambiguous surfaces, *Philos. Trans. R. Soc. London A* **332**(1623), 1–47 (1990)

- 23.20 A. Jepson, D.J. Heeger: A fast subspace algorithm for recovering rigid motion, Proc. IEEE Workshop on Visual Motion (Princeton 1991) pp.124–131
- 23.21 C. Fermüller, Y. Aloimonos: Algorithmic independent instability of structure from motion, Proc. 5th Eur. Conf. Comput. Vis. (Freiburg 1998)
- 23.22 K. Daniilidis, M. Spetsakis: Understanding noise sensitivity in structure from motion. In: *Visual Navigation*, ed. by Y. Aloimonos. (Lawrence Erlbaum, Hillsdale 1996), pp.61–88
- 23.23 S.R. Soatto Brockett: Optimal structure from motion: Local ambiguities and global estimates, IEEE Conf. Comput. Vis. Pattern Recog. (Santa Barbara 1998)
- 23.24 J. Oliensis: A new structure-from-motion ambiguity, IEEE Trans. Pattern Anal. Mach. Intell. **22**, 685–700 (1999)
- 23.25 Y. Ma, K. Huang, R. Vidal, J. Kosecka, S. Sastry: Rank conditions of the multiple view matrix, Int. J. Comput. Vis. **59**(2), 115–137 (2004)
- 23.26 Y. Ma, S. Soatto, J. Kosecka, S. Sastry: *An Invitation to 3-D Vision* (Springer, Berlin, Heidelberg 2003)
- 23.27 W. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon: Bundle adjustment for structure from motion (Springer Verlag 2000) pp. 298–375
- 23.28 M. Lourakis, A. Argyros: The design and implementation of a generic sparse bundle adjustment software package based on the Levenberg–Marquard method. Technical Report 340, ICS/FORTH (2004)
- 23.29 S. Teller, M. Antone, Z. Bodnar, M. Bosse, S. Coorg: Calibrated, registered images of an extended urban area, Int. Conf. Comput. Vis. Pattern Recogn., Vol.1 (Kanai 2001) pp. 813–820
- 23.30 E. Trucco, A. Verri: *Introductory Techniques for 3-D Computer Vision* (Prentice Hall, Upper Saddle River 1998)
- 23.31 S.S. Intille, A.F. Bobick: Disparity-space images and large occlusion stereo, ECCV **2**, 179–186 (1994)
- 23.32 R. Szeliski, D. Scharstein: Sampling the disparity space image, IEEE Trans. Pattern Anal. Mach. Intell. **26**(3), 419–425 (2004)
- 23.33 R. Yang, M. Pollefeys, G. Welch: Dealing with textureless regions and specular highlights: A progressive space carving scheme using a novel photo-consistency measure, Proc. Int. Conf. Comput. Vis. (2003)
- 23.34 X. Zabulis, A. Patterson, K. Daniilidis: Digitizing archaeological excavations from multiple monocular views, 5th Int. Conf. 3-D Digital Imag. Mod. (2005)
- 23.35 R.T. Collins: A space-sweep approach to true multi-image matching, IEEE Conf. Comput. Vis. Pattern Recog. (San Fransisco 1996) pp. 358–363
- 23.36 T. Kanade, M. Okutomi: A stereo matching algorithm with an adaptive window: Theory and experiment, IEEE Trans. Pattern Anal. Mach. Intell. **16**(9), 920–932 (1994)
- 23.37 D. Scharstein, R. Szeliski: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms, Int. J. Comput. Vis. **47**(1/2/3), 7–42 (2002)
- 23.38 H. Hirschmuller: Stereo vision in structured environments by consistent semi-global matching, Comput. Vis. Pattern Recog. **02**, 2386–2393 (2006)
- 23.39 O. Veksler: Stereo correspondence by dynamic programming on a tree, Comput. Vis. Pattern Recog. **2**, 384–390 (2005)
- 23.40 S. Roy, I. Cox: A maximum-flow formulation of the N-camera stereo correspondence problem, Proc. Int. Conf. Comput. Vis. (1998)
- 23.41 V. Kolmogorov, R. Zabih: Computing visual correspondence with occlusions using graph cuts, Int. Conf. Comput. Vis. **02**, 508 (2001)
- 23.42 H.-Y. Shum, J. Sun, N.-N. Zheng: Stereo matching using belief propagation, IEEE Trans. Pattern Anal. Mach. Intell. **25**, 787–800 (2003)
- 23.43 L. Zhang, S.M. Seitz: Estimating optimal parameters for mrf stereo from a single image pair, IEEE Trans. Pattern Anal. Mach. Intell. **29**(2), 331–342 (2007)
- 23.44 P.F. Felzenszwalb, D.P. Huttenlocher: Efficient belief propagation for early vision, Comput. Vis. Pattern Recog. **01**, 261–268 (2004)
- 23.45 H. Hirschmuller: Accurate and efficient stereo processing by semi-global matching and mutual information, Comput. Vis. Pattern Recog. **2**, 807–814 (2005)
- 23.46 S.M. Seitz, B. Curless, J. Diebel, D. Scharstein, R. Szeliski: A comparison and evaluation of multi-view stereo reconstruction algorithms, Comput. Vis. Pattern Recog. **1**, 519–528 (2006)
- 23.47 C.R. Dyer: Volumetric scene reconstruction from multiple views. In: *Foundations of Image Understanding*, ed. by L. Davis (Kluwer, Boston 2001) pp. 469–489
- 23.48 D.A. Forsyth, J. Ponce: *Computer Vision: A Modern Approach*, Prentice Hall Professional Technical Reference (Prentice Hall, Upper Saddle River 2002)
- 23.49 L. Fei Fei, R. Fergus, A. Torralba: Recognizing and learning object categories, Short course given at CVPR 2007 (2007)
- 23.50 A. Pinz: Object categorization, Foundations and Trends in Computer Graphics and Vision **1**(4), 255–353 (2005)
- 23.51 A. Guzman: Decomposition of a visual scene into three-dimensional bodies. In: *Automatic Interpretation and Classification of Images*, ed. by A. Grasselli (Academic, New York 1965)
- 23.52 T.O. Binford: Visual perception by computer, Proc. IEEE Conf. Syst. Contr. (Miami 1971)
- 23.53 R. Brooks: *Model-Based Computer Vision* (Kluwer Academic, Dordrecht 1984)
- 23.54 D. Marr, K. Nishihara: Representation and recognition of the spatial organization of three-dimensional shapes, Proc. R. Soc. London B **200**, 269–294 (1978)

- 23.55 D. Marr: *Vision* (Freeman, New York 1990)
- 23.56 O.D. Faugeras, M. Hebert: The representation, recognition, and localization of 3D objects, *Int. J. Rob. Res.* **5**(3), 27–52 (1986)
- 23.57 R.C. Bolles, P. Horaud: 3dpo: A three-dimensional part orientation system, *Int. J. Robot. Res.* **5**(3), 3–26 (1986)
- 23.58 I. Biederman: Human image understanding: recent research and a theory, *Comput. Vis. Graphics Image Process.* **32**, 29–73 (1985)
- 23.59 R. Mohan, R. Nevatia: Perceptual organization for scene segmentation and description, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(6), 616–635 (1992)
- 23.60 A. Zisserman, J.L. Mundy, D.A. Forsyth, J. Liu, N. Pillow, C. Rothwell, S. Utcke: Class-based grouping in perspective images, *Int. Conf. Comput. Vis.* (1995) pp.183–188
- 23.61 R.C. Nelson, A. Selinger: Large-scale tests of a keyed, appearance-based 3d object recognition system, *Vis. Res. special issue on computational vision* **38**, 15–16 (1998)
- 23.62 M.J. Tarr, H.H. Bülthoff: Image-based object recognition in man, monkey and machine. In: *Object Recognition in Man, Monkey, and Machine*, ed. by M.J. Tarr, H. H. Bülthoff (MIT Press, Cambridge 1998) pp.1–20
- 23.63 M. Turk, A. Pentland: Eigenfaces for recognition, *J. Cognit. Neurosci.* **3**, 71–86 (1991)
- 23.64 T. Poggio, S. Edelman: A neural network that learns to recognize three-dimensional object, *Nature* **343**, 263–266 (1990)
- 23.65 H. Murase, S.K. Nayar: Visual learning and recognition of 3-d objects from appearance, *Int. J. Comput. Vis.* **14**(1), 5–24 (1995)
- 23.66 R.P.N. Rao, D.H. Ballard: Object indexing using an iconic sparse distributed memory. Tech. Rep. TR559, University of Rochester (1995)
- 23.67 M.A. Fischler, R.A. Elschlager: The representation and matching of pictorial structure, *IEEE Trans. Comput.* **22**, 67–92 (1973)
- 23.68 R.M. Haralick, L.G. Shapiro: *Computer and Robot Vision* (Addison-Wesley, Boston 1992)
- 23.69 T. Lindeberg: On the axiomatic foundations of linear scale-space: Combining semi-group structure with causality vs. scale invariance. In: *Gaussian Scale-Space Theory: Proc. PhD School on Scale-Space Theory* (Kluwer Academic, Dordrecht 1994)
- 23.70 D.G. Lowe: Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
- 23.71 C. Schmid, R. Mohr: Local grayvalue invariants for image retrieval, *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(5), 530–534 (1997)
- 23.72 K. Mikolajczyk, C. Schmid: An affine invariant interest point detector. In: *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, ed. by A. Heyden, G. Sparr, P. Johansen, M. Nielsen (Springer, Berlin, Heidelberg 2002) pp.128–142
- 23.73 J. Matas, O. Chum, M. Urban, T. Pajdla: Robust wide baseline stereo from maximally stable extremal regions, *Br. Mach. Vis. Conf.* (2002)
- 23.74 K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, L. Van Gool: A comparison of affine region detectors, *Int. J. Comput. Vis.* **65**(1/2), 43–72 (2005)
- 23.75 H.P. Moravec: Towards automatic visual obstacle avoidance, *IJCAI* (1977) p. 584
- 23.76 C. Harris, M.J. Stephens: A combined corner and edge detector, *Alvey Vision Conference* (1988) pp.147–152
- 23.77 W. Foerstner: On the geometric precision of digital correlation, *Int. Arch. Photogram. Rem. Sens.* (1982)
- 23.78 G. Granlund, J. Bigun: Optimal orientation detection of linear symmetry, *Proc. IEEE 1st Int. Conf. Comput. Vis.* (1987)
- 23.79 D.G. Lowe: Object recognition from local scale-invariant features, *Proc. Int. Conf. Comput. Vis., Corfu* (1999) pp.1150–1157
- 23.80 T. Lindeberg: Feature detection with automatic scale selection, *Int. J. Comput. Vis.* **30**(2), 79–116 (1998)
- 23.81 R. Sedgewick: *Algorithms (2nd ed.)* (Addison-Wesley, Boston 1988)
- 23.82 T. Kadir, J.M. Brady: Scale, salience and image description, *Int. J. Comput. Vis.* **45**, 83–105 (2001)
- 23.83 S.S. Smith, J.M. Brady: Susan – a new approach to low level image processing, *Int. J. Comput. Vis.* **23**, 45–78 (1997)
- 23.84 B. Leibe, B. Schiele, A. Leonardis: Combined object categorization and segmentation with an implicit shape model, *Europ. Conf. Comp. Vision* (2004)
- 23.85 B. Schiele, J.L. Crowley: Recognition without correspondence using multidimensional receptive field histograms, *Int. J. Comput. Vis.* **36**(1), 31–50 (2000)
- 23.86 S. Agarwal, D. Roth: Learning a sparse representation for object detection, *Proc. 7th Eur. Conf. Comput. Vis., Vol. 4* (2002) pp.113–130
- 23.87 M.J. Swain, D.H. Ballard: Color indexing, *Int. J. Comput. Vis.* **7**, 11–32 (1991)
- 23.88 H. Schneiderman, T. Kanade: A statistical method for 3d object detection applied to faces and cars, *IEEE Conf. Comput. Vis. Pattern Recog.* (2000)
- 23.89 O. Linde, T. Lindeberg: Object recognition using composed receptive field histograms of higher dimensionality, *Proc. Int. Conf. Pattern Recog.* (2004)
- 23.90 J.J. Koenderink, A.J. Van Doorn: The structure of locally orderless images, *Int. J. Comput. Vis.* **31**(2–3), 159–168 (1999)
- 23.91 Y. Rubner, C. Tomasi: *Perceptual Metrics for Image Database Navigation* (Kluwer Academic, Dordrecht 2000)

- 23.92 J. Sivic, A. Zisserman: Video Google: A text retrieval approach to object matching in videos, Proc. 9th Int. Conf. Comput. Vis. (Nice 2003) pp.1470–1477
- 23.93 R. Baeza-Yates, B. Ribeiro-Neto: *Modern Information Retrieval* (Addison Wesley, Reading 1999)
- 23.94 D. Nister, H. Stewenius: Scalable recognition with a vocabulary tree, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog. (2006) pp.2161–2168
- 23.95 K. Grauman, T. Darrell: Approximate correspondences in high dimensions, Adv. Neural Inform. Proc. Syst **19**, 505–512 (2007)
- 23.96 J. Beis, D. Lowe: Shape indexing using approximate nearest-neighbor search in highdimensional spaces
- 23.97 P. Indyk, R. Motwani: Approximate nearest neighbors: towards removing the curse of dimensionality, Proc. 30th Ann. ACM Symp. Theory Comput. (1998) pp. 604–613
- 23.98 O. Drbohlav, D. Omercevic, A. Leonardis: High-dimensional feature matching: Employing the concept of meaningful nearest neighbors, Proc. 11th Int. Conf. Comput. Vis. (2007), in press
- 23.99 P.F. Felzenszwalb, D.P. Huttenlocher: Pictorial structures for object recognition, Int. J. Comput. Vis. **61**(1), 55–79 (2005)
- 23.100 R. Fergus, P. Perona, A. Zisserman: Weakly supervised scale-invariant learning of models for visual recognition, Int. J. Comput. Vis. (2005)
- 23.101 J. Kosecka, F. Li: Vision based Markov localization, ICRA (2004)
- 23.102 J. Wolf, W. Burgard, H. Burkhardt: Using an image retrieval system for visionbased mobile robot localization (2002)
- 23.103 I. Ulrich, I. Nourbakhsh: Appearance-based place recognition for topological localization, Proc. ICRA, Vol. 2 (2000) pp.1023–1029
- 23.104 A. Davison, D. Murray: Simultaneous localisation and map-building using active vision, IEEE Trans. Pattern Anal. Mach. Intell. **24**, 865–880 (2002)
- 23.105 R. Bajcsy: Active perception, Proc. IEEE **76**, 996–1005 (1988)
- 23.106 F.T. Ramos, B. Upcroft, S. Kumar, H.F. Durrant-Whyte: A Bayesian approach for place recognition, Int. Joint Conf. Artif. Intell. Workshop on Reasoning with Uncertainty in Robotics (RUR-05) (2005)
- 23.107 O. Faugeras: *Three-dimensional Computer Vision* (MIT Press, Cambridge 1993)
- 23.108 A. Akbarzadeh, J.-M. Frahm, P. Mordohai, B. Clipp, C. Engels, D. Gallup, P. Merell, M. Phels, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewenius, R. Yang, G. Welch, H. Towles, D. Nister, M. Pollefeys: Towards urban 3D reconstruction from video, Third Int. Symp. on 3D Data Processing, Visualization, and Transmission (2006)
- 23.109 D. Nister: Preemptive ransac for live structure and motion estimation, Proc. Int. Conf. Comput. Vis. (2003) pp.199–206
- 23.110 P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J.-M. Frahm, R. Yang, D. Nister, M. Pollefeys: Real-time visibility-based fusion of depth maps, Int. Conf. Comput. Vis. (2007)

24. Visual Servoing and Visual Tracking

François Chaumette, Seth Hutchinson

This chapter introduces visual servo control, using computer vision data in the servo loop to control the motion of a robot. We first describe the basic techniques that are by now well established in the field. We give a general overview of the formulation of the visual servo control problem, and describe the two archetypal visual servo control schemes: image-based and position-based visual servo control. We then discuss performance and stability issues that pertain to these two schemes, motivating advanced techniques. Of the many advanced techniques that have been developed, we discuss 2.5-D, hybrid, partitioned, and switched approaches. Having covered a variety of control schemes, we conclude by turning briefly to the problems of target tracking and controlling motion directly in the joint space.

24.1 The Basic Components of Visual Servoing	564
24.2 Image-Based Visual Servo	565
24.2.1 The Interaction Matrix	565
24.2.2 Approximating the Interaction Matrix	565

24.2.3 A Geometrical Interpretation of IBVS	567
24.2.4 Stability Analysis	568
24.2.5 IBVS with a Stereo Vision System	570
24.2.6 IBVS with Cylindrical Coordinates of Image Points	570
24.2.7 IBVS with Other Geometrical Features	571
24.2.8 Direct Estimation	571
24.3 Position-Based Visual Servo	572
24.4 Advanced Approaches	574
24.4.1 Hybrid VS	574
24.4.2 Partitioned VS	576
24.5 Performance Optimization and Planning	577
24.5.1 Optimal Control and Redundancy Framework	577
24.5.2 Switching Schemes	577
24.5.3 Feature Trajectory Planning	578
24.6 Estimation of 3-D Parameters	578
24.7 Target Tracking	579
24.8 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space	580
24.9 Conclusions	581
References	582

Visual servo (VS) control refers to the use of computer vision data to control the motion of a robot. The vision data may be acquired from a camera that is mounted directly on a robot manipulator or on a mobile robot, in which case motion of the robot induces camera motion, or the camera can be fixed in the workspace so that it can observe the robot motion from a stationary configuration. Other configurations can be considered, such as for instance several cameras mounted on pan-

tilt heads observing the robot motion. The mathematical development of all these cases is similar, and in this chapter we will focus primarily on the former, so-called *eye-in-hand*, case.

Visual servo control relies on techniques from image processing, computer vision, and control theory. In the present chapter, we will deal primarily with the issues from control theory, making connections to previous chapters when appropriate.

24.1 The Basic Components of Visual Servoing

The aim of all vision-based control schemes is to minimize an error $e(t)$, which is typically defined by

$$e(t) = s[m(t), a] - s^*. \quad (24.1)$$

This formulation is quite general, and it encompasses a wide variety approaches, as we will see below. The parameters in (24.1) are defined as follows. The vector $m(t)$ is a set of image measurements (e.g., the image coordinates of interest points, or the parameters of a set of image segments). These image measurements are used to compute a vector of k visual features, $s(m(t), a)$, in which a is a set of parameters that represent potential additional knowledge about the system (e.g., coarse camera intrinsic parameters or three-dimensional model of objects). The vector s^* contains the desired values of the features.

For now, we consider the case of a fixed goal pose and a motionless target, i. e., s^* is constant, and changes in s depend only on camera motion. Further, we consider here the case of controlling the motion of a camera with six degrees of freedom (e.g., a camera attached to the end effector of a six-degree-of-freedom arm). We will treat more general cases in later sections.

Visual servoing schemes mainly differ in the way that s is designed. In Sects. 24.2 and 24.3, we describe classical approaches, including image-based visual servo control (IBVS), in which s consists of a set of features that are immediately available in the image data, and position-based visual servo control (PBVS), in which s consists of a set of three-dimensional (3-D) parameters, which must be estimated from image measurements. We also present in Sect. 24.4 several more-advanced methods.

Once s is selected, the design of the control scheme can be quite simple. Perhaps the most straightforward approach is to design a velocity controller. To do this, we require the relationship between the time variation of s and the camera velocity. Let the spatial velocity of the camera be denoted by $v_c = (v_c, \omega_c)$ where v_c is the instantaneous linear velocity of the origin of the camera frame and ω_c is the instantaneous angular velocity of the camera frame. The relationship between \dot{s} and v_c is given by

$$\dot{s} = L_s v_c, \quad (24.2)$$

in which $L_s \in \mathbb{R}^{k \times 6}$ is called the *interaction matrix* related to s [24.1]. The term *feature Jacobian* is also used somewhat interchangeably in the visual servo literature [24.2], but in the present chapter we will use this

last term to relate the time variation of the features to the joints velocity (Sect. 24.8).

Using (24.1) and (24.2) we immediately obtain the relationship between camera velocity and the time variation of the error:

$$\dot{e} = L_e v_c, \quad (24.3)$$

where $L_e = L_s$. Considering v_c as the input to the robot controller, and if we would like, for instance, to try to ensure an exponential decoupled decrease of the error (i. e., $\dot{e} = -\lambda e$), we obtain using (24.3)

$$v_c = -\lambda L_e^+ e, \quad (24.4)$$

where $L_e^+ \in \mathbb{R}^{6 \times k}$ is chosen as the Moore–Penrose pseudo-inverse of L_e , that is, $L_e^+ = (L_e^\top L_e)^{-1} L_e^\top$ when L_e is of full rank 6. This choice allows $\|\dot{e} - \lambda L_e L_e^+ e\|$ and $\|v_c\|$ to be minimal. When $k = 6$, if $\det L_e \neq 0$ it is possible to invert L_e , giving the control $v_c = -\lambda L_e^{-1} e$.

In real visual servo systems, it is impossible to know perfectly in practice either L_e or L_e^+ . So an approximation or an estimation of one of these two matrices must be realized. In the sequel, we denote both the pseudo-inverse of the approximation of the interaction matrix and the approximation of the pseudo-inverse of the interaction matrix by the symbol \widehat{L}_e^+ . Using this notation, the control law is in fact

$$v_c = -\lambda \widehat{L}_e^+ e = -\lambda \widehat{L}_s^+ (s - s^*). \quad (24.5)$$

Closing the loop and assuming that the robot controller is able to realize perfectly v_c , that is inserting (24.5) into (24.3), we obtain

$$\dot{e} = -\lambda L_e \widehat{L}_e^+ e. \quad (24.6)$$

This equation characterizes the real behavior of the closed-loop system, which is different from the specified one ($\dot{e} = -\lambda e$) as soon as $L_e \widehat{L}_e^+ \neq I_6$, where I_6 is the 6×6 identity matrix. It is also at the basis of the stability analysis of the system using Lyapunov theory.

What we have presented above is the basic design implemented by most visual servo controllers. All that remains is to fill in the details. How should s be chosen? What then is the form of L_s ? How should we estimate \widehat{L}_e^+ ? What are the performance characteristics of the resulting closed-loop system? These questions are addressed in the remainder of the chapter. We first describe the two basic approaches, IBVS and PBVS, whose principles were proposed more than 20 years ago [24.3]. We then present more-recent approaches that have improved their performance.

24.2 Image-Based Visual Servo

Traditional image-based control schemes [24.3, 4] use the image-plane coordinates of a set of points to define the set s . The image measurements \mathbf{m} are usually the pixel coordinates of the set of image points (although this is not the only possible choice), and the parameters \mathbf{a} in the definition of $s = s(\mathbf{m}, \mathbf{a})$ in (24.1) are nothing but the camera intrinsic parameters to go from image measurements expressed in pixels to the features.

24.2.1 The Interaction Matrix

For a 3-D point with coordinates $X = (X, Y, Z)$ in the camera frame, which projects in the image as a two-dimensional (2-D) point with coordinates $\mathbf{x} = (x, y)$, we have

$$\begin{cases} x &= X/Z = (u - c_u)/f\alpha \\ y &= Y/Z = (v - c_v)/f \end{cases}, \quad (24.7)$$

where $\mathbf{m} = (u, v)$ gives the coordinates of the image point expressed in pixel units, and $\mathbf{a} = (c_u, c_v, f, \alpha)$ is the set of camera intrinsic parameters as defined in Chap. 23: c_u and c_v are the coordinates of the principal point, f is the focal length, and α is the ratio of the pixel dimensions. The intrinsic parameter β defined in Chap. 23 has been assumed to be 0 here. In this case, we take $\mathbf{s} = \mathbf{x} = (x, y)$, the image plane coordinates of the point. The details of the imaging geometry and perspective projection can be found in many computer vision texts, including [24.5, 6].

Taking the time derivative of the projection equations (24.7), we obtain

$$\begin{cases} \dot{x} &= \dot{X}/Z - X\dot{Z}/Z^2 = (\dot{X} - x\dot{Z})/Z \\ \dot{y} &= \dot{Y}/Z - Y\dot{Z}/Z^2 = (\dot{Y} - y\dot{Z})/Z \end{cases}. \quad (24.8)$$

We can relate the velocity of the 3-D point to the camera spatial velocity using the well-known equation

$$\dot{\mathbf{X}} = -\mathbf{v}_c - \boldsymbol{\omega}_c \times \mathbf{X} \Leftrightarrow \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \end{cases}. \quad (24.9)$$

Inserting (24.9) into (24.8), grouping terms, and using (24.7) we obtain

$$\begin{cases} \dot{x} = -v_x/Z + xv_z/Z + xy\omega_x - (1+x^2)\omega_y + y\omega_z \\ \dot{y} = -v_y/Z + yv_z/Z + (1+y^2)\omega_x - xy\omega_y - x\omega_z \end{cases}, \quad (24.10)$$

which can be written

$$\dot{\mathbf{x}} = \mathbf{L}_x \mathbf{v}_c, \quad (24.11)$$

where the interaction matrix \mathbf{L}_x is given by

$$\mathbf{L}_x = \begin{pmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{pmatrix}. \quad (24.12)$$

In the matrix \mathbf{L}_x , the value Z is the depth of the point relative to the camera frame. Therefore, any control scheme that uses this form of the interaction matrix must estimate or approximate the value of Z . Similarly, the camera intrinsic parameters are involved in the computation of x and y . Thus \mathbf{L}_x^+ cannot be *directly* used in (24.4), and an estimation or an approximation $\widehat{\mathbf{L}}_x^+$ must be used, as in (24.5). We discuss this in more detail below.

To control the six degrees of freedom, at least three points are necessary (i. e., we require $k \geq 6$). If we use the feature vector $\mathbf{x} = (x_1, x_2, x_3)$, by merely stacking interaction matrices for three points we obtain

$$\mathbf{L}_x = \begin{pmatrix} \mathbf{L}_{x_1} \\ \mathbf{L}_{x_2} \\ \mathbf{L}_{x_3} \end{pmatrix}.$$

In this case, there will exist some configurations for which \mathbf{L}_x is singular [24.7]. Furthermore, there exist four distinct camera poses for which $\mathbf{e} = \mathbf{0}$, i. e., four global minima exist, and it is impossible to differentiate them [24.8]. For these reasons, more than three points are usually considered.

24.2.2 Approximating the Interaction Matrix

There are several choices available for constructing the estimate $\widehat{\mathbf{L}}_e^+$ to be used in the control law. One popular scheme is of course to choose $\widehat{\mathbf{L}}_e^+ = \mathbf{L}_e^+$ if $\mathbf{L}_e = \mathbf{L}_x$ is known, that is if the current depth Z of each point is available [24.2]. In practice, these parameters must be estimated at each iteration of the control scheme. The basic IBVS methods use classical pose-estimation methods (see Chap. 23 and the beginning of Sect. 24.3). Another popular approach is to choose $\widehat{\mathbf{L}}_e^+ = \mathbf{L}_{e^*}^+$ where \mathbf{L}_{e^*} is the value of \mathbf{L}_e for the desired position $\mathbf{e} = \mathbf{e}^* = \mathbf{0}$ [24.1]. In this case, $\widehat{\mathbf{L}}_e^+$ is constant, and only the desired depth of each point has to be set, which means no varying

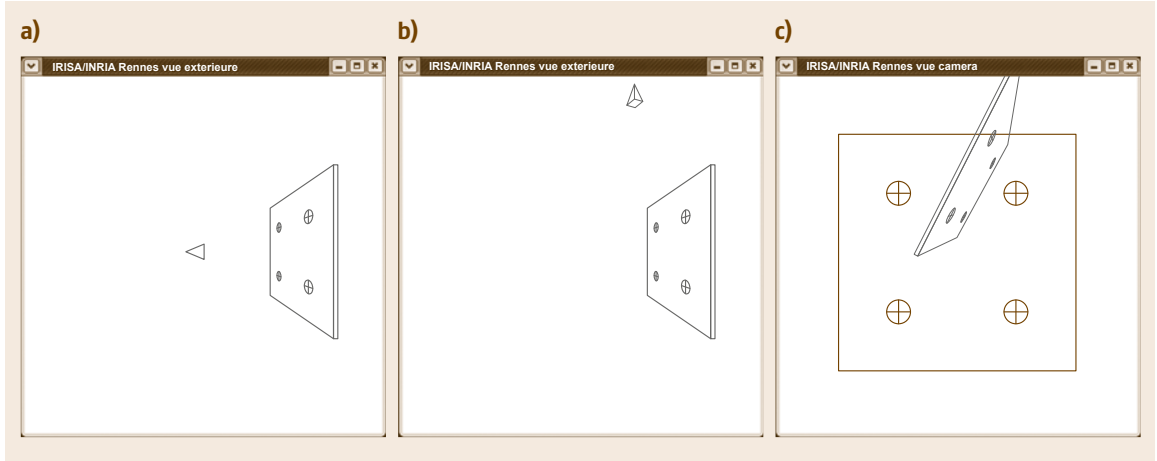


Fig. 24.1a–c Example of positioning task: **(a)** the desired camera pose with respect to a simple target, **(b)** the initial camera pose, and **(c)** the corresponding initial and desired image of the target

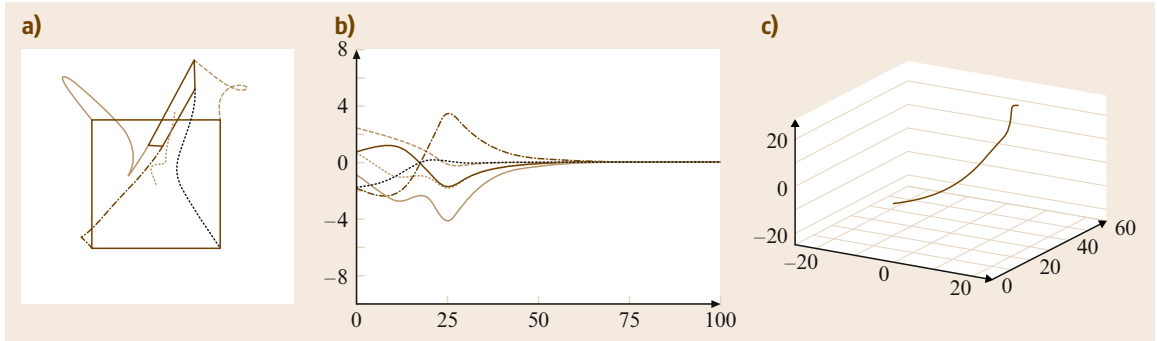


Fig. 24.2a–c System behavior using $s = (x_1, y_1, \dots, x_4, y_4)$ and $\widehat{L}_e^+ = L_e^+$: **(a)** image point trajectories including the trajectory of the center of the square, which is not used in the control scheme, **(b)** v_c components (cm/s and deg/s) computed at each iteration of the control scheme, and **(c)** the 3-D trajectory of the camera optical center expressed in \mathcal{R}_{c^*} (cm)

3-D parameters have to be estimated during the visual servo. Finally, the choice $\widehat{L}_e^+ = (L_e/2 + L_{e^*}/2)^+$ has recently been proposed [24.9]. Since L_e is involved in this method, the current depth of each point also has to be available.

We illustrate the behavior of these control schemes with an example. The goal is to position the camera so that it observes a square as a centered square in the image (see Fig. 24.1). We define s to include the x and y coordinates of the four points forming the square. Note that the initial camera pose has been selected far away from the desired pose, particularly with regard to the rotational motions, which are known to be the most problematic for IBVS. In the simulations presented in

the following, no noise or modeling errors have been introduced in order to allow comparison of different behaviors in perfect conditions.

The results obtained by using $\widehat{L}_e^+ = L_e^+$ are given in Fig. 24.2. Note that despite the large displacement that is required the system converges. However, neither the behavior in the image, nor the computed camera velocity components, nor the 3-D trajectory of the camera present desirable properties far from the convergence (i. e., for the first 30 or so iterations).

The results obtained using $\widehat{L}_e^+ = L_e^+$ are given in Fig. 24.3. In this case, the trajectories of the points in the image are almost straight lines, but the behavior induced in the camera frame is even less satisfactory than

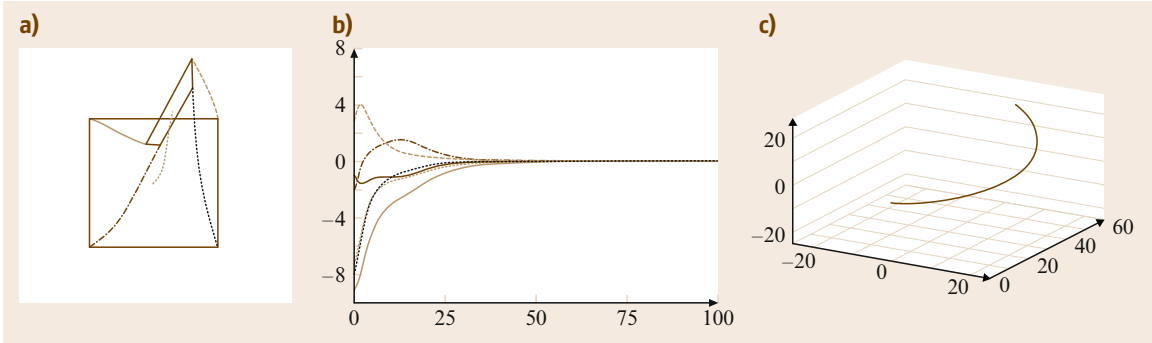


Fig. 24.3a-c System behavior using $s = (x_1, y_1, \dots, x_4, y_4)$ and $\widehat{L}_e^+ = L_e^+$

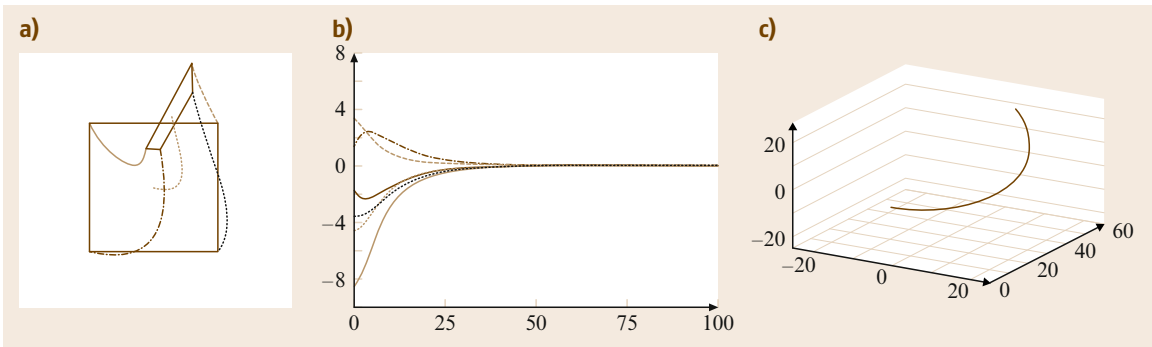


Fig. 24.4a-c System behavior using $s = (x_1, y_1, \dots, x_4, y_4)$ and $\widehat{L}_e^+ = (L_e/2 + L_e^*/2)^+$

for the case of $\widehat{L}_e^+ = L_e^+$. The large camera velocities at the beginning of the servo indicate that the condition number of L_e^+ is high at the start of the trajectory, and the camera trajectory is far from a straight line.

The choice $\widehat{L}_e^+ = (L_e/2 + L_e^*/2)^+$ provides good performance in practice. Indeed, as can be seen in Fig. 24.4, the camera velocity components do not include large oscillations, and provide a smooth trajectory in both the image and in 3-D space.

24.2.3 A Geometrical Interpretation of IBVS

It is quite easy to provide a geometric interpretation of the behavior of the control schemes defined above. The example illustrated in Fig. 24.5 corresponds to a pure rotation around the optical axis from the initial configuration (shown in blue) to the desired configuration of four coplanar points parallel to the image plane (shown in red).

As explained above, using L_e^+ in the control scheme attempts to ensure an exponential decrease of the error e . This means that, when x and y image point coordi-

nates compose this error, the points' trajectories in the image follow straight lines from their initial to their desired positions, when this is possible. This leads to the image motion plotted in green in the figure. The camera motion to realize this image motion can be easily deduced and is indeed composed of a rotational motion around the optical axis, but combined with a retreating translational motion along the optical axis [24.10]. This unexpected motion is due to the choice of the features, and to the coupling between the third and sixth columns in the interaction matrix. If the rotation between the initial and desired configurations is very large, this phenomenon is amplified, and leads to a particular case for a rotation of π radians where no rotational motion at all will be induced by the control scheme [24.11]. On the other hand, when the rotation is small, this phenomenon almost disappears. To conclude, the behavior is locally satisfactory (i. e., when the error is small), but it can be unsatisfactory when the error is large. As we will see below, these results are consistent with the local asymptotic stability results that can be obtained for IBVS.

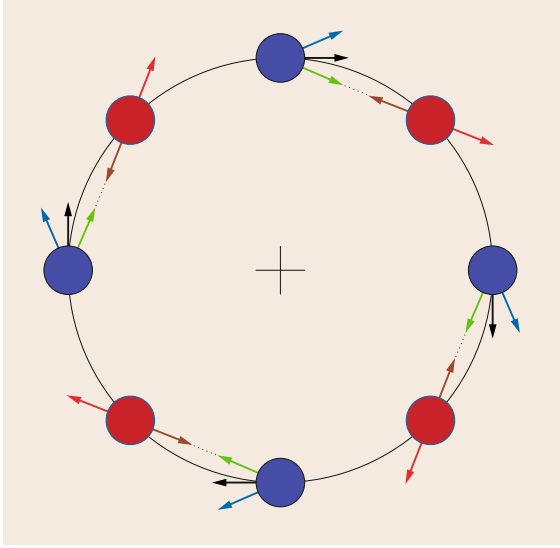


Fig. 24.5 Geometrical interpretation of IBVS: going from the blue position to the red one. In *red*, image motion when L_e^+ is used in the control scheme; in *blue*, when $L_{e^*}^+$ is used; and in *black* when $(L_e/2 + L_{e^*}/2)^+$ is used (see the text for more details)

If instead we use $L_{e^*}^+$ in the control scheme, the image motion generated can easily be shown to be the blue one plotted in Fig. 24.5. Indeed, if we consider the same control scheme as before but starting from s^* to reach s , we obtain

$$v_c = -\lambda L_{e^*}^+(s^* - s),$$

which again induces straight-line trajectories from the red points to the blue ones, causing the image motion plotted in brown. Going back to our problem, the control scheme computes a camera velocity that is exactly the opposite one

$$v_c = -\lambda L_e^+(s - s^*),$$

and thus generates the image motion plotted in red at the red points. Transformed at the blue points, the camera velocity generates the blue image motion and corresponds once again to a rotational motion around the optical axis, combined now with an unexpected forward motion along the optical axis. The same analysis as before can be done, as for large or small errors. We can add that, as soon as the error decreases significantly, both control schemes get closer, and tend to the same one (since $L_e = L_{e^*}$ when $e = e^*$) with a nice behavior characterized with the image motion plotted in black and

a camera motion composed of only a rotation around the optical axis when the error tends towards zero.

If we instead use $\widehat{L}_e^+ = (L_e/2 + L_{e^*}/2)^+$, it is intuitively clear that considering the mean of L_e and L_{e^*} generates the image motion plotted in black, even when the error is large. In all cases but the rotation around π radians, the camera motion is now a pure rotation around the optical axis, without any unexpected translational motion.

24.2.4 Stability Analysis

We now consider the fundamental issues related to the stability of IBVS. To assess the stability of the closed-loop visual servo systems, we will use Lyapunov analysis. In particular, consider the candidate Lyapunov function defined by the squared error norm $\mathcal{L} = \frac{1}{2} \|e(t)\|^2$, whose derivative is given by

$$\begin{aligned} \dot{\mathcal{L}} &= e^\top \dot{e} \\ &= -\lambda e^\top L_e \widehat{L}_e^+ e \end{aligned}$$

since \dot{e} is given by (24.6). The global asymptotic stability of the system is thus obtained when the following sufficient condition is satisfied:

$$L_e \widehat{L}_e^+ > 0. \quad (24.13)$$

If the number of features is equal to the number of camera degrees of freedom (i. e., $k = 6$), and if the features are chosen and the control scheme designed so that L_e and \widehat{L}_e^+ are of full rank 6, then the condition (24.13) is satisfied if the approximations involved in \widehat{L}_e^+ are not too coarse.

As discussed above, for most IBVS approaches we have $k > 6$. Therefore the condition (24.13) can never be ensured since $L_e \widehat{L}_e^+ \in \mathbb{R}^{k \times k}$ is at most of rank 6, and thus $L_e \widehat{L}_e^+$ has a nontrivial null space. In this case, configurations such that $e \in \ker \widehat{L}_e^+$ correspond to local minima. Reaching such a local minimum is illustrated in Fig. 24.6. As can be seen in Fig. 24.6d, each component of e has a nice exponential decrease with the same convergence speed, causing straight-line trajectories to be realized in the image, but the error reached is not exactly zero, and it is clear from Fig. 24.6c that the system has been attracted to a local minimum far away from the desired configuration. Thus, only local asymptotic stability can be obtained for IBVS.

To study local asymptotic stability when $k > 6$, let us first define a new error e' with $e' = \widehat{L}_e^+ e$. The time

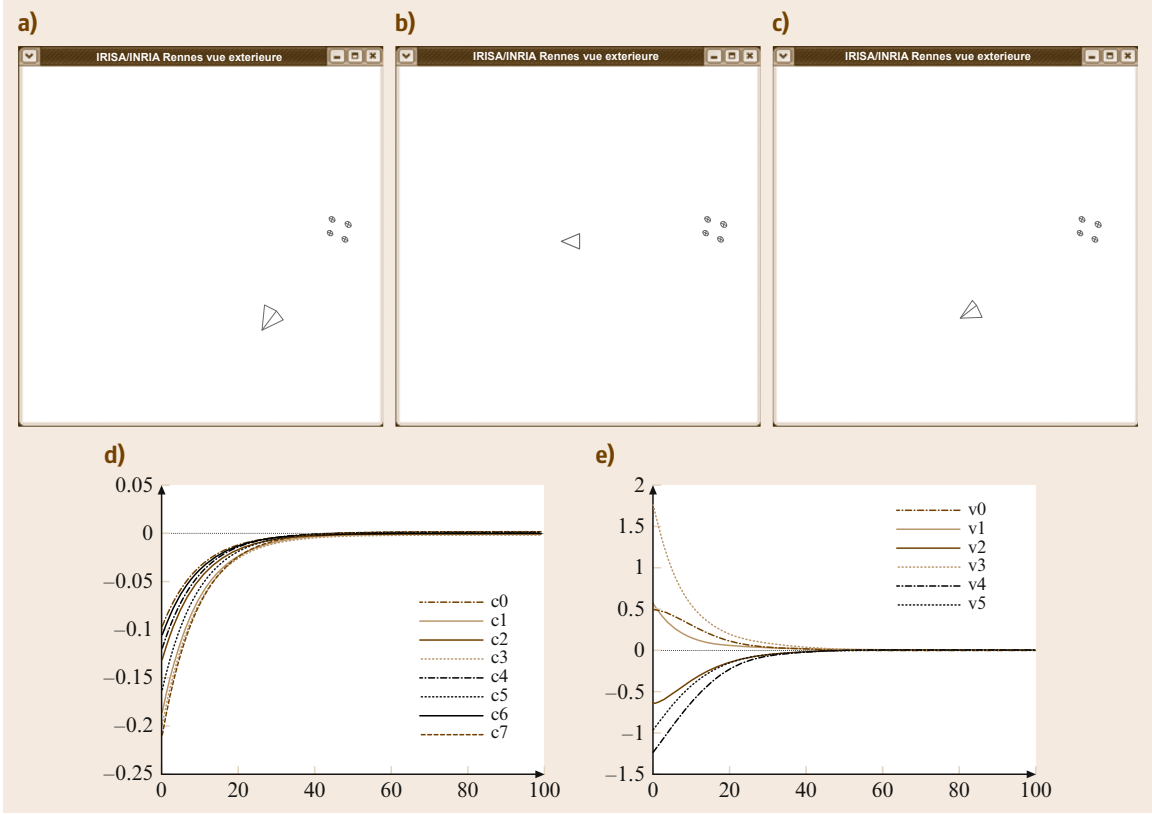


Fig. 24.6a–e Reaching a local minimum using $s = (x_1, y_1, \dots, x_4, y_4)$ and $\widehat{L}_e^+ = L_e^+$: (a) the initial configuration, (b) the desired one, (c) the configuration reached after the convergence of the control scheme, (d) the evolution of the error e at each iteration of the control scheme, and (e) the evolution of the six components of the camera velocity v_c

derivative of this error is given by

$$\begin{aligned} \dot{e}' &= \widehat{L}_e^+ \dot{e} + \widehat{L}_e^+ e \\ &= (\widehat{L}_e^+ L_e + O) v_c, \end{aligned}$$

where $O \in \mathbb{R}^{6 \times 6}$ is equal to 0 when $e = 0$, whatever the choice of \widehat{L}_e^+ [24.12]. Using the control scheme (24.5), we obtain

$$\dot{e}' = -\lambda (\widehat{L}_e^+ L_e + O) e',$$

which is known to be locally asymptotically stable in a neighborhood of $e = e^* = 0$ if

$$\widehat{L}_e^+ L_e > 0, \quad (24.14)$$

where $\widehat{L}_e^+ L_e \in \mathbb{R}^{6 \times 6}$. Indeed, only the linearized system $\dot{e}' = -\lambda \widehat{L}_e^+ L_e e'$ has to be considered if we are interested in the local asymptotic stability [24.13].

Once again, if the features are chosen and the control scheme designed so that L_e and \widehat{L}_e^+ are of full rank 6, then condition (24.14) is ensured if the approximations involved in \widehat{L}_e^+ are not too coarse.

To end the demonstration of local asymptotic stability, we must show that there does not exist any configuration $e \neq e^*$ such that $e \in \ker \widehat{L}_e^+$ in a small neighborhood of e^* and in a small neighborhood of the corresponding pose p^* . Such configurations correspond to local minima where $v_c = 0$ and $e \neq e^*$. If such a pose p would exist, it is possible to restrict the neighborhood around p^* so that there exists a camera velocity v to reach p^* from p . This camera velocity would imply a variation of the error $\dot{e} = L_e v$. However, such a variation cannot belong to $\ker \widehat{L}_e^+$ since $\widehat{L}_e^+ L_e > 0$. Therefore, we have $v_c = 0$ if and only if $\dot{e} = 0$, i.e., $e = e^*$, in a neighborhood of p^* .

Even though local asymptotic stability can be ensured when $k > 6$, we recall that global asymptotic stability cannot be ensured. For instance, as illustrated in Fig. 24.6, there may exist local minima corresponding to configurations where $e \in \ker \widehat{L_e^+}$, which are outside of the neighborhood considered above. Determining the size of the neighborhood in which stability and the convergence are ensured is still an open issue, even if this neighborhood is surprisingly quite large in practice.

24.2.5 IBVS with a Stereo Vision System

It is straightforward to extend the IBVS approach to a multicamera system. If a stereo vision system is used, and a 3-D point is visible in both left and right images (see Fig. 24.7), it is possible to use as visual features

$$s = x_s = (x_l, x_r) = (x_l, y_l, x_r, y_r)$$

i. e., to represent the point by just stacking in s the x and y coordinates of the observed point in the left and right images [24.14]. However, care must be taken when constructing the corresponding interaction matrix since the form given in (24.11) is expressed in either the left or right camera frame. More precisely, we have

$$\begin{cases} \dot{x}_l = L_{x_l} v_l, \\ \dot{x}_r = L_{x_r} v_r, \end{cases}$$

where v_l and v_r are the spatial velocity of the left and right camera, respectively, and where the analytical form of L_{x_l} and L_{x_r} are given by (24.12).

By choosing a sensor frame rigidly linked to the stereo vision system, we obtain

$$\dot{x}_s = \begin{pmatrix} \dot{x}_l \\ \dot{x}_r \end{pmatrix} = L_{x_s} v_s,$$

where the interaction matrix related to x_s can be determined using the spatial motion transform matrix V

defined in Chap. 1 to transform velocities expressed in the left or right cameras frames to the sensor frame. We recall that V is given by

$$V = \begin{pmatrix} R & [t]_\times R \\ 0 & R \end{pmatrix}, \quad (24.15)$$

where $[t]_\times$ is the skew-symmetric matrix associated to the vector t and where $(R, t) \in SE(3)$ is the rigid-body transformation from the camera to the sensor frame. The numerical values for these matrices are directly obtained from the calibration step of the stereo vision system. Using this equation, we obtain

$$L_{x_s} = \begin{pmatrix} L_{x_l}^1 V_s \\ L_{x_r}^r V_s \end{pmatrix}.$$

Note that $L_{x_s} \in \mathbb{R}^{4 \times 6}$ is always of rank 3 because of the epipolar constraint that links the perspective projection of a 3-D point in a stereo vision system (see Fig. 24.7). Another simple interpretation is that a 3-D point is represented by three independent parameters, which makes it impossible to find more than three independent parameters using any sensor observing that point.

To control the six degrees of freedom of the system, it is necessary to consider at least three points, as the rank of the interaction matrix considering only two points is 5.

Using a stereo vision system, since the 3-D coordinates of any point observed in both images can be easily estimated by a simple triangulation process it is possible and quite natural to use these 3-D coordinates in the features set s . Such an approach would be, strictly speaking, a position-based approach, since it would require 3-D parameters in s .

24.2.6 IBVS with Cylindrical Coordinates of Image Points

In the previous sections, we have considered the Cartesian coordinates of images points. As proposed in [24.15], it may be interesting to use the cylindrical coordinates (ρ, θ) of the image points instead of their Cartesian coordinates (x, y) . They are given by:

$$\rho = \sqrt{x^2 + y^2}, \quad \theta = \arctan \frac{y}{x}$$

from which we deduce

$$\dot{\rho} = (x\dot{x} + y\dot{y})/\rho, \quad \dot{\theta} = (x\dot{y} - y\dot{x})/\rho^2.$$

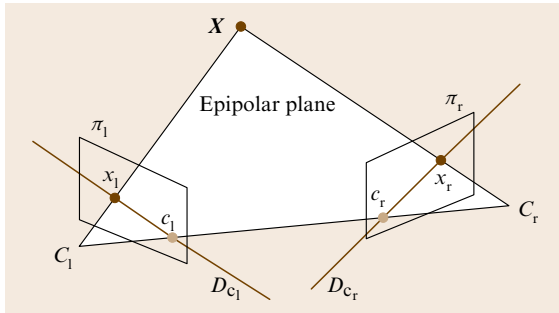


Fig. 24.7 A stereo vision system

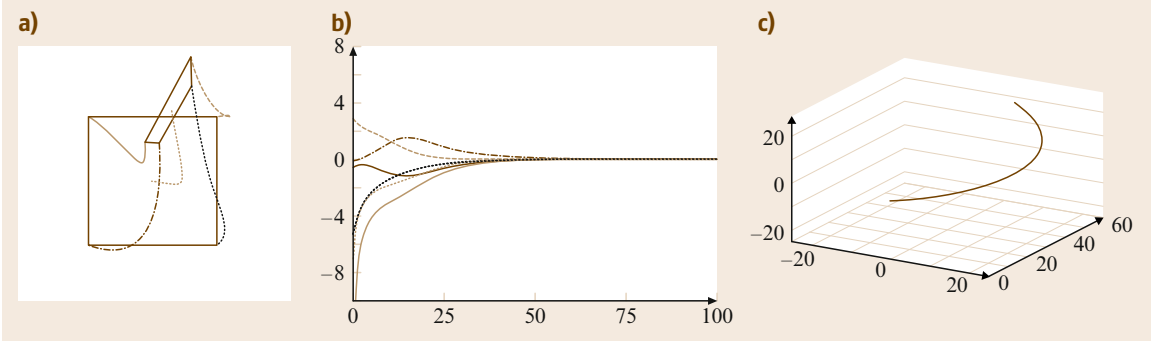


Fig. 24.8a-c System behavior using $s = (\rho_1, \theta_1, \dots, \rho_4, \theta_4)$ and $\widehat{L}_e^+ = L_e^+$

Using (24.11) and then substituting x by $\rho \cos \theta$ and y by $\rho \sin \theta$, we obtain immediately

$$L_\rho = \begin{pmatrix} -\frac{c}{\rho Z} & -\frac{s}{\rho Z} & \frac{\rho}{Z} & (1 + \rho^2)s & -(1 + \rho^2)c & 0 \end{pmatrix},$$

$$L_\theta = \begin{pmatrix} \frac{s}{\rho Z} & -\frac{c}{\rho Z} & 0 & \frac{c}{\rho} & \frac{s}{\rho} & -1 \end{pmatrix},$$

where $c = \cos \theta$ and $s = \sin \theta$.

The behavior obtained in that case is quite satisfactory. Indeed, by considering the same example as before, and using now $s = (\rho_1, \theta_1, \dots, \rho_4, \theta_4)$ and $\widehat{L}_e^+ = L_e^+$ (that is, a constant matrix), the system behavior shown on Fig. 24.8 has the same nice properties as when using $\widehat{L}_e^+ = (L_e/2 + L_{e^*}/2)^+$ with $s = (x_1, y_1, \dots, x_4, y_4)$. If we go back to the example depicted in Fig. 24.5, the behavior obtained will be also the expected one thanks to the decoupling between the third and sixth columns of the interaction matrix.

24.2.7 IBVS with Other Geometrical Features

In the previous sections, we have only considered image point coordinates in s . Other geometrical primitives can of course be used. There are several reasons to do so. First, the scene observed by the camera cannot always be described merely by a collection of points, in which case the image processing provides other types of measurements, such as a set of straight lines or the contours of an object. Second, richer geometric primitives may ameliorate the decoupling and linearizing issues that motivate the design of partitioned systems. Finally, the robotic task to be achieved may be expressed in terms of virtual linkages (or fixtures) between the camera and the observed objects [24.16, 17], sometimes expressed directly by constraints between primitives, such as point-

to-line [24.18] (which means that an observed point must lie on a specified line).

For the first point, it is possible to determine the interaction matrix related to the perspective projection of a large class of geometrical primitives, such as segments, straight lines, spheres, circles, and cylinders. The results are given in [24.1] and [24.16]. Recently, the analytical form of the interaction matrix related to any image moments corresponding to planar objects has been computed. This makes it possible to consider planar objects of any shape [24.19]. If a collection of points is measured in the image, moments can also be used [24.20]. In both cases, moments allow the use of intuitive geometrical features, such as the center of gravity or the orientation of an object. By selecting an adequate combination of moments, it is then possible to determine partitioned systems with good decoupling and linearizing properties [24.19, 20].

Note that, for all these features (geometrical primitives, moments), the depth of the primitive or of the object considered appears in the coefficients of the interaction matrix related to the translational degrees of freedom, as was the case for the image points. An estimation of this depth is thus generally still necessary. A few exceptions occur, using for instance an adequate normalization of moments, which allow in some particular cases to make only the constant desired depth appear in the interaction matrix [24.20].

24.2.8 Direct Estimation

In the previous sections, we have focused on the analytical form of the interaction matrix. It is also possible to estimate its numerical value directly using either an offline learning step, or an online estimation scheme.

All the methods proposed to estimate the interaction matrix numerically rely on the observation of a variation of the features due to a known or measured camera motion. More precisely, if we measure a feature's variation Δs due to a camera motion Δv_c , we have from (24.2):

$$L_s \Delta v_c = \Delta s,$$

which provides k equations while we have $k \times 6$ unknown values in L_s . Using a set of N independent camera motions with $N > 6$, it is thus possible to estimate L_s by solving

$$L_s A = B,$$

where the columns of $A \in \mathbb{R}^{6 \times N}$ and $B \in \mathbb{R}^{k \times N}$ are, respectively, formed from the set of camera motions and the set of corresponding features variations. The least-square solution is of course given by

$$\hat{L}_s = BA^+.$$
 (24.16)

Methods based on neural networks have also been developed to estimate L_s [24.21, 22]. It is also possible to estimate the numerical value of L_s^+ directly, which in practice provides a better behavior [24.23]. In this case, the basic relation is

$$L_s^+ \Delta s = \Delta v_c,$$

which provides six equations. Using a set of N measurements, with $N > k$, we now obtain

$$\hat{L}_s^+ = AB^+.$$
 (24.17)

In the first case (24.16), the six columns of L_s are estimated by solving six linear systems, while in the second case (24.17), the k columns of L_s^+ are estimated by solving k linear systems, which explains the difference in the results.

Estimating the interaction matrix online can be viewed as an optimization problem, and consequently a number of researchers have investigated approaches that derive from optimization methods. These methods typically discretize the system equation (24.2), and use an iterative updating scheme to refine the estimate of \hat{L}_s at each stage. One such online and iterative formulation uses the Broyden update rule given by [24.24, 25]:

$$\hat{L}_s(t+1) = \hat{L}_s(t) + \frac{\alpha}{\Delta v_c^T \Delta v_c} [\Delta x - \hat{L}_s(t) \Delta v_c] \Delta v_c^T,$$

where α defines the update speed. This method has been generalized to the case of moving objects in [24.26].

The main interest of using such numerical estimations in the control scheme is that it avoids all the modeling and calibration steps. It is particularly useful when using features whose interaction matrix is not available in analytical form. For instance, in [24.27], the main eigenvalues of the principal component analysis of an image have been considered in a visual servoing scheme. The drawback of these methods is that no theoretical stability and robustness analysis can be made.

24.3 Position-Based Visual Servo

Position-based control schemes (PBVS) [24.3, 28, 29] use the pose of the camera with respect to some reference coordinate frame to define s . Computing this pose from a set of measurements in one image necessitates the camera intrinsic parameters and the 3-D model of the object observed to be known. This classic computer vision problem is called the 3-D localization problem. While this problem is beyond the scope of the present chapter, many solutions have been presented in the literature [24.30, 31] and its basic principles are recalled in Chap. 23.

It is then typical to define s in terms of the parameterization used to represent the camera pose. Note that the parameters a involved in the definition (24.1) of s are now the camera intrinsic parameters and the 3-D model of the object.

It is convenient to consider three coordinate frames: the current camera frame \mathcal{F}_c , the desired camera frame

\mathcal{F}_{c^*} , and a reference frame \mathcal{F}_0 attached to the object. We adopt here the standard notation of using a leading superscript to denote the frame with respect to which a set of coordinates is defined. Thus, the coordinate vectors ${}^c t_0$ and ${}^{c^*} t_0$ give the coordinates of the origin of the object frame expressed relative to the current camera frame, and relative to the desired camera frame, respectively. Furthermore, let $R = {}^c R_c$ be the rotation matrix that gives the orientation of the current camera frame relative to the desired frame.

We can define s to be $(t, \theta u)$, in which t is a translation vector, and θu gives the angle/axis parameterization for the rotation. We now discuss two choices for t , and give the corresponding control laws.

If t is defined relative to the object frame \mathcal{F}_0 , we obtain $s = ({}^c t_0, \theta u)$, $s^* = ({}^{c^*} t_0, 0)$, and $e = ({}^c t_0 - {}^{c^*} t_0, \theta u)$. In this case, the interaction matrix related to e is given

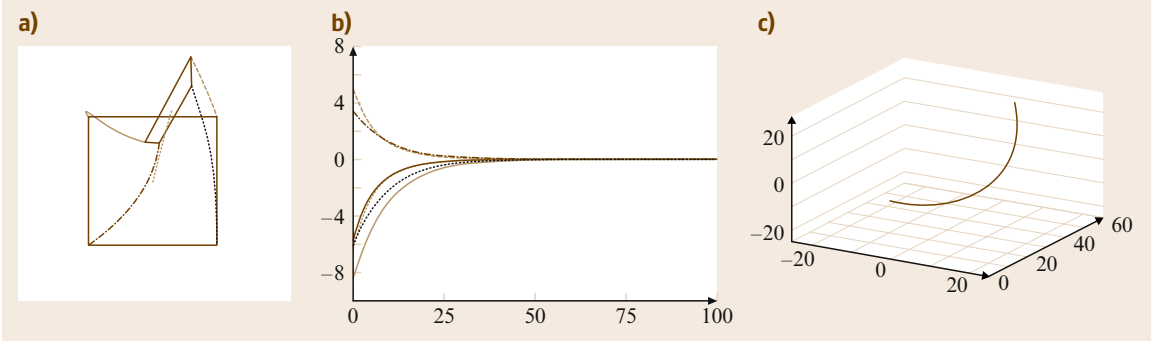


Fig. 24.9a-c System behavior using $s = ({}^c t_0, \theta u)$

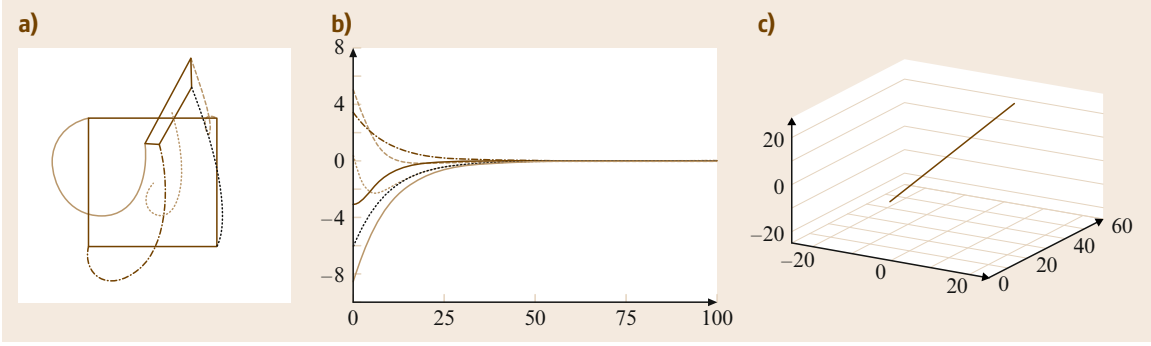


Fig. 24.10a-c System behavior using $s = ({}^c {}^* t_c, \theta u)$

by

$$L_e = \begin{pmatrix} -I_3 & [{}^c t_0]_\times \\ \mathbf{0} & L_{\theta u} \end{pmatrix}, \quad (24.18)$$

in which I_3 is the 3×3 identity matrix and $L_{\theta u}$ is given by [24.32]

$$L_{\theta u} = I_3 - \frac{\theta}{2} [u]_\times + \left(1 - \frac{\text{sinc } \theta}{\text{sinc}^2 \frac{\theta}{2}} \right) [u]_\times^2, \quad (24.19)$$

where $\text{sinc } x$ is the sinus cardinal defined such that $x \text{ sinc } x = \sin x$ and $\text{sinc } 0 = 1$.

Following the development in Sect. 24.1, we obtain the control scheme

$$v_c = -\lambda \widehat{L_e}^{-1} e$$

since the dimension k of s is six, that is, the number of camera degrees of freedom. By setting

$$\widehat{L_e}^{-1} = \begin{pmatrix} -I_3 & [{}^c t_0]_\times L_{\theta u}^{-1} \\ \mathbf{0} & L_{\theta u}^{-1} \end{pmatrix}, \quad (24.20)$$

we obtain after simple developments:

$$\begin{cases} v_c &= -\lambda [{}^c {}^* t_c - {}^c t_0] + [{}^c t_0]_\times \theta u \\ R_c &= -\lambda \theta u \end{cases}. \quad (24.21)$$

since $L_{\theta u}$ is such that $L_{\theta u}^{-1} \theta u = \theta u$.

Ideally, that is, if the pose parameters are perfectly estimated, the behavior of e will be the expected one ($\dot{e} = -\lambda e$). The choice of e causes the rotational motion to follow a geodesic with an exponential decreasing speed and causes the translational parameters involved in s to decrease with the same speed. This explains the nice exponential decrease of the camera velocity components in Fig. 24.9. Furthermore, the trajectory in the image of the origin of the object frame follows a pure straight line (here the center of the four points has been selected as this origin). On the other hand, the camera trajectory does not follow a straight line.

Another PBVS scheme can be designed by using $s = ({}^c {}^* t_c, \theta u)$. In this case, we have $s^* = \mathbf{0}$, $e = s$, and

$$L_e = \begin{pmatrix} R & \mathbf{0} \\ \mathbf{0} & L_{\theta u} \end{pmatrix}. \quad (24.22)$$

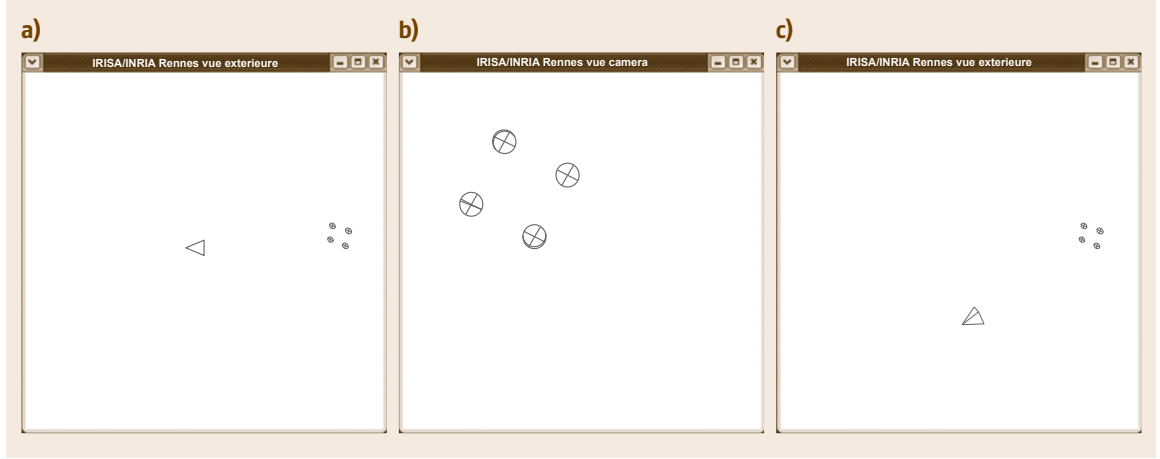


Fig. 24.11a–c Two different camera poses (a, c) that provide almost the same image of four coplanar points (b)

Note the decoupling between translational and rotational motions, which allows us to obtain a simple control scheme

$$\begin{cases} \mathbf{v}_c = -\lambda \mathbf{R}^\top \mathbf{c}^* \mathbf{t}_c \\ \boldsymbol{\omega}_c = -\lambda \theta \mathbf{u} \end{cases} \quad (24.23)$$

In this case, as can be seen in Fig. 24.10, if the pose parameters involved in (24.23) are estimated perfectly, the camera trajectory is a pure straight line, while the image trajectories are less satisfactory than before. Some particular configurations can even be found so that some points leave the camera field of view.

The stability properties of PBVS seem quite attractive. Since $\mathbf{L}_{\theta \mathbf{u}}$ given in (24.19) is nonsingular when $\theta \neq 2k\pi$, we obtain from (24.13) the global asymptotic stability of the system since $\mathbf{L}_e \widehat{\mathbf{L}}_e^{-1} = \mathbf{I}_6$, under the strong hypothesis that all the pose parameters are perfect. This is true for both methods presented above, since the interactions matrices given in (24.18)

and (24.22) are full rank when $\mathbf{L}_{\theta \mathbf{u}}$ is nonsingular.

With regard to robustness, feedback is computed using *estimated* quantities that are a function of the image measurements and the system calibration parameters. For the first method presented in Sect. 24.3 (the analysis for the second method is analogous), the interaction matrix given in (24.18) corresponds to perfectly estimated pose parameters, while the real one is unknown since the estimated pose parameters may be biased due to calibration errors, or inaccurate and unstable due to noise [24.11]. The true positivity condition (24.13) should in fact be written:

$$\widehat{\mathbf{L}}_e \widehat{\mathbf{L}}_e^{-1} > 0, \quad (24.24)$$

where $\widehat{\mathbf{L}}_e^{-1}$ is given by (24.20) but where \mathbf{L}_e is unknown, and not given by (24.18). Indeed, even small errors in computing the points position in the image can lead to pose errors that can impact significantly the accuracy and the stability of the system (see Fig. 24.11).

24.4 Advanced Approaches

24.4.1 Hybrid VS

Suppose we have access to a clever control law for $\boldsymbol{\omega}_c$, such as the one used in PBVS [see (24.21) or (24.23)]:

$$\boldsymbol{\omega}_c = -\lambda \theta \mathbf{u}. \quad (24.25)$$

How could we use this in conjunction with traditional IBVS?

Considering a feature vector \mathbf{s}_t and an error \mathbf{e}_t devoted to control the translational degrees of freedom, we can partition the interaction matrix as follows

$$\begin{aligned} \dot{\mathbf{s}}_t &= \mathbf{L}_{\mathbf{s}_t} \mathbf{v}_c \\ &= \begin{pmatrix} \mathbf{L}_v \mathbf{L}_\omega \end{pmatrix} \begin{pmatrix} \mathbf{v}_c \\ \boldsymbol{\omega}_c \end{pmatrix} \\ &= \mathbf{L}_v \mathbf{v}_c + \mathbf{L}_\omega \boldsymbol{\omega}_c. \end{aligned}$$

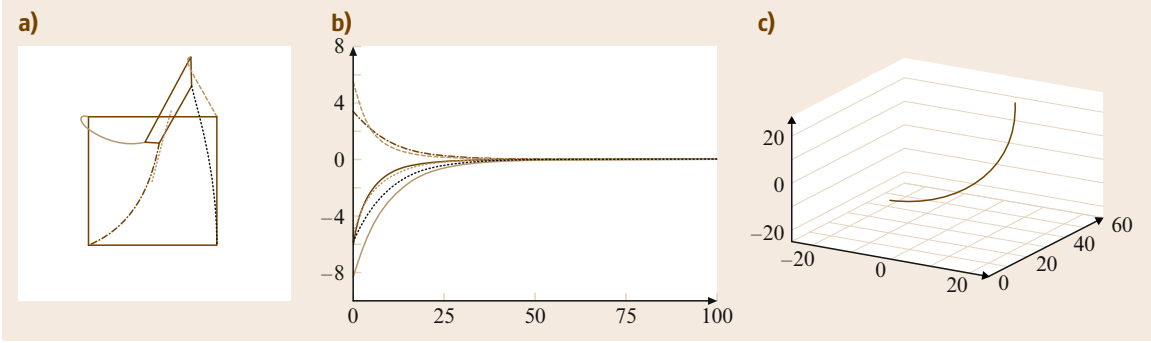


Fig. 24.12a–c System behavior using $s = (x_g, \log(Z_g), \theta u)$

Now, setting $\dot{e}_t = -\lambda e_t$, we can solve for the desired translational control input as

$$\begin{aligned} -\lambda e_t &= \dot{e}_t = \dot{s}_t = L_v v_c + L_\omega \omega_c, \\ \Rightarrow v_c &= -L_v^+ (\lambda e_t + L_\omega \omega_c). \end{aligned} \quad (24.26)$$

We can think of the quantity $(\lambda e_t + L_\omega \omega_c)$ as a modified error term, one that combines the original error with the error that would be induced by the rotational motion due to ω_c . The translational control input $v_c = -L_v^+ (\lambda e_t + L_\omega \omega_c)$ will drive this error to zero. The method known as 2.5-D visual servo [24.32] was the first to exploit such a partitioning in combining IBVS and PBVS. More precisely, in [24.32], s_t has been selected as the coordinates of an image point, and the logarithm of its depth, so that L_v is a triangular always invertible matrix. More precisely, we have $s_t = (x, \log Z)$, $s_t^* = (x^*, \log Z^*)$, $e_t = (x - x^*, \log \rho_Z)$ where $\rho_Z = Z/Z^*$, and

$$L_v = \frac{1}{Z^* \rho_Z} \begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1 \end{pmatrix}$$

$$L_\omega = \begin{pmatrix} xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0 \end{pmatrix}.$$

Note that the ratio ρ_Z can be obtained directly from the partial pose estimation algorithm that will be described in Sect. 24.6.

If we come back to the usual global representation of visual servo control schemes, we have $e = (e_t, \theta u)$ and L_e given by

$$L_e = \begin{pmatrix} L_v & L_\omega \\ 0 & L_{\theta u} \end{pmatrix},$$

from which we immediately obtain the control law (24.25) and (24.26) by applying (24.5).

The behavior obtained using this choice for s_t is shown in Fig. 24.12. Here, the point that has been considered in s_t is the center of gravity x_g of the target. We note the image trajectory of that point, which is a straight line as expected, and the nice decreasing of the camera velocity components, which makes this scheme very near from the first PBVS one.

As for stability, it is clear that this scheme is globally asymptotically stable in perfect conditions. Furthermore, thanks to the triangular form of the interaction matrix L_e , it is possible to analyze the stability of this scheme in the presence of calibration errors using the partial pose-estimation algorithm that will be described in Sect. 24.6 [24.33]. Finally, the only unknown constant parameter involved in this scheme, that is Z^* , can be estimated online using adaptive techniques [24.34].

Other hybrid schemes can be designed, for instance, in [24.35], the third component of s_t is different and has been selected so that all the target points remain in the camera field of view as far as possible. Another example has been proposed in [24.36]. In that case, s is selected as $s = ({}^c t_c, x_g, \theta u_z)$ which provides with a block-triangular interaction matrix of the form:

$$L_e = \begin{pmatrix} R & 0 \\ L'_v & L'_\omega \end{pmatrix}$$

where L'_v and L'_ω can easily be computed. This scheme is such that, under perfect conditions, the camera trajectory is a straight line (since ${}^c t_c$ is a part of s), and the image trajectory of the center of gravity of the object is also a straight line (since x_g is also a part of s). The translational camera degrees of freedom are de-

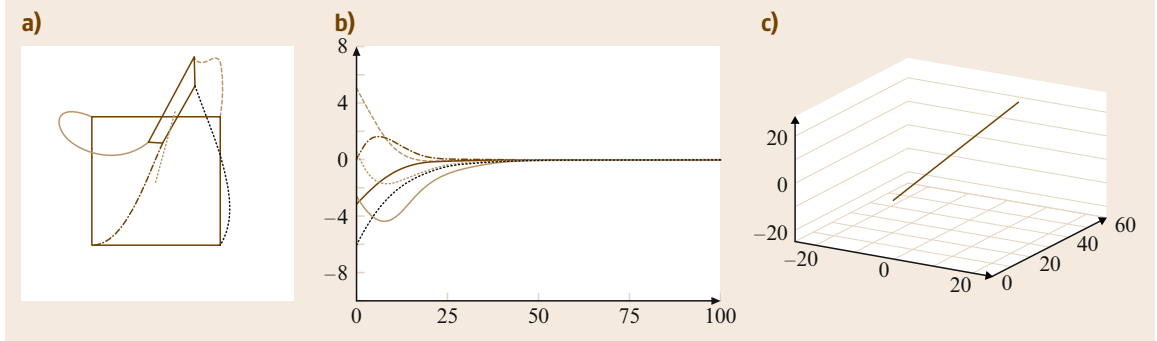


Fig. 24.13a–c System behavior using $s = (c^* t_c, x_g, \theta u_z)$

voted to realize the 3-D straight line, while the rotational camera degrees of freedom are devoted to realize the 2-D straight line and compensate also the 2-D motion of x_g due to the translational motion. As can be seen in Fig. 24.13, this scheme is particularly satisfactory in practice.

Finally, it is possible to combine 2-D and 3-D features in different ways. For instance, in [24.37], it has been proposed to use in s the 2-D homogeneous coordinates of a set of image points expressed in pixels multiplied by their corresponding depth: $s = (u_1 Z_1, v_1 Z_1, Z_1, \dots, u_n Z_n, v_n Z_n, Z_n)$. As for classical IBVS, we obtain in this case a set of redundant features, since at least three points have to be used to control the six camera degrees of freedom (here $k \geq 9$). However, it has been demonstrated in [24.38] that this selection of redundant features is free of attractive local minima.

24.4.2 Partitioned VS

The hybrid visual servo schemes described above have been designed to decouple the rotational motions from the translational ones by selecting adequate visual features defined in part in 2-D, and in part in 3-D (which is why they have been called 2.5-D visual servoing). This work has inspired some researchers to find features that exhibit similar decoupling properties but using only features expressed directly in the image. More precisely, the goal is to find six features such that each is related to only one degree of freedom (in which case the interaction matrix is a diagonal matrix), the Grail is to find a diagonal interaction matrix whose elements are constant, as near as possible to the identity matrix, leading to a pure, direct, and simple linear control problem.

The first work in this area partitioned the interaction matrix to isolate motion related to the optic axis [24.10]. Indeed, whatever the choice of s , we have

$$\begin{aligned}\dot{s} &= L_s v_c \\ &= L_{xy} v_{xy} + L_z v_z \\ &= \dot{s}_{xy} + \dot{s}_z\end{aligned}$$

in which L_{xy} includes the first, second, fourth, and fifth columns of L_s , and L_z includes the third and sixth columns of L_s . Similarly, $v_{xy} = (v_x, v_y, \omega_x, \omega_y)$ and $v_z = (v_z, \omega_z)$. Here, $\dot{s}_z = L_z v_z$ gives the component of \dot{s} due to the camera motion along and rotation about the optic axis, while $\dot{s}_{xy} = L_{xy} v_{xy}$ gives the component of \dot{s} due to velocity along and rotation about the camera x and y axes.

Proceeding as above, by setting $\dot{e} = -\lambda e$ we obtain

$$-\lambda e = \dot{e} = \dot{s} = L_{xy} v_{xy} + L_z v_z,$$

which leads to

$$v_{xy} = -L_{xy}^+ [\lambda e(t) + L_z v_z].$$

As before, we can consider $[\lambda e(t) + L_z v_z]$ as a modified error that incorporates the original error while taking into account the error that will be induced by v_z .

Given this result, all that remains is to choose s and v_z . As for basic IBVS, the coordinates of a collection of image points can be used in s , while two new image features can be defined to determine v_z .

- Define α , with $0 \leq \alpha < 2\pi$, as the angle between the horizontal axis of the image plane and the directed line segment joining two feature points.

It is clear that α is closely related to the rotation around the optic axis.

- Define σ^2 to be the area of the polygon defined by these points. Similarly, σ^2 is closely related to the translation along the optic axis.

Using these features, v_z has been defined in [24.10] as

$$\begin{cases} v_z &= \lambda_{v_z} \ln \frac{\sigma^*}{\sigma}, \\ \omega_z &= \lambda_{\omega_z} (\alpha^* - \alpha). \end{cases}$$

24.5 Performance Optimization and Planning

In some sense, partitioned methods represent an effort to optimize system performance by assigning distinct features and controllers to individual degrees of freedom. In this way, the designer performs a sort of offline optimization when allocating controllers to degrees of freedom. It is also possible to explicitly design controllers that optimize various system performance measures. We describe a few of these in this section.

24.5.1 Optimal Control and Redundancy Framework

An example of such an approach is given in [24.39] and [24.40], in which linear quadratic Gaussian (LQG) control design is used to choose gains that minimize a linear combination of state and control input. This approach explicitly balances the trade-off between tracking errors (since the controller attempts to drive $s - s^*$ to zero) and robot motion. A similar control approach is proposed in [24.41] where joint limit avoidance is considered simultaneously with the positioning task.

It is also possible to formulate optimality criteria that explicitly express the observability of robot motion in the image. For example, the singular value decomposition of the interaction matrix reveals which degrees of freedom are most apparent and can thus be easily controlled, while the condition number of the interaction matrix gives a kind of global measure of the visibility of motion. This concept has been called resolvability in [24.42] and motion perceptibility in [24.43]. By selecting features and designing controllers that maximize these measures, either along specific degrees of freedom or globally, the performance of the visual servo system can be improved.

The constraints considered to design the control scheme using the optimal control approach may be contradictory in some cases, leading the system to fail due to local minima in the objective function to be minimized. For example, it may happen that the motion produced to move away from a robot joint limit is exactly the opposite of the motion produced to near the desired pose, which results in a zero global motion. To avoid

this potential problem, it is possible to use the gradient projection method, which is classical in robotics. Applying this method to visual servoing has been proposed in [24.1] and [24.17]. The approach consists of projecting the secondary constraints e_s onto the null space of the vision-based task e so that they have no effect on the regulation of e to 0:

$$e_g = \widehat{L}_e^+ e + P_e e_s,$$

where e_g is the new global task considered and $P_e = (I_6 - \widehat{L}_e^+ \widehat{L}_e)$ is such that $\widehat{L}_e P_e e_s = 0, \forall e_s$. Avoiding the robot joint limits using this approach has been presented in [24.44]. However, when the vision-based task constrains all the camera degrees of freedom, the secondary constraints cannot be considered since, when \widehat{L}_e is of full rank 6, we have $P_e e_s = 0, \forall e_s$. In this case, it is necessary to insert the constraints into a global objective function, such as navigation functions that are free of local minima [24.45, 46].

24.5.2 Switching Schemes

The partitioned methods described previously attempt to optimize performance by assigning individual controllers to specific degrees of freedom. Another way to use multiple controllers to optimize performance is to design switching schemes that select at each moment in time which controller to use based on criteria to be optimized.

A simple switching controller can be designed using an IBVS and a PBVS controller as follows [24.47]. Let the system begin by using the IBVS controller. Consider the Lyapunov function for the PBVS controller given by $\mathcal{L} = \frac{1}{2} \|e(t)\|^2$, with $e(t) = ({}^c t_0 - {}^c t_0, \theta u)$. If at any time the value of this Lyapunov function exceeds a threshold γ_P , the system switches to the PBVS controller. While using the PBVS controller, if at any time the value of the Lyapunov function for the IBVS controller exceeds a threshold, $\mathcal{L} = \frac{1}{2} \|e(t)\|^2 > \gamma_I$, the system switches to the IBVS controller. With this scheme, when the Lyapunov function for a particular controller exceeds

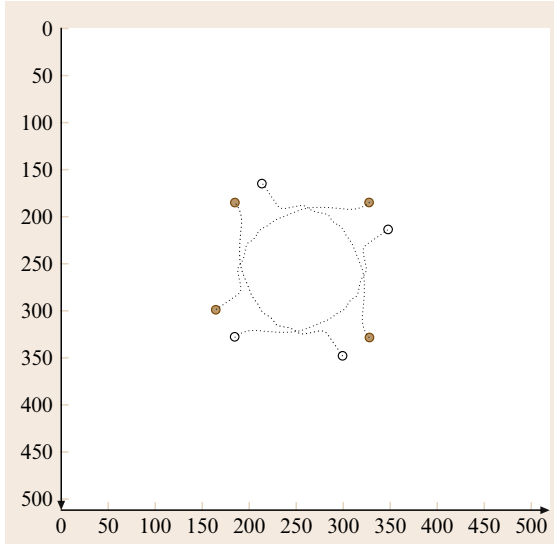


Fig. 24.14 Image feature trajectories for a rotation of 160° about the optical axis using a switched control scheme (initial point positions in blue, and desired points position in red)

a threshold, that controller is invoked, which in turn reduces the value of the corresponding Lyapunov function. If the switching thresholds are selected appropriately, the system is able to exploit the relative advantages of IBVS and PBVS, while avoiding their shortcomings.

An example of such a system is shown in Fig. 24.14 for the case of a rotation by 160° about the optical axis. Note that the system begins in IBVS mode and the features initially move on straight lines toward their goal positions in the image. However, as the camera retreats, the system switches to PBVS, which allows the camera to reach its desired position by combining a rotational motion around its optic axis and a forward translational motion, producing the circular trajectories observed in the image.

24.6 Estimation of 3-D Parameters

All the control schemes described in the previous sections use 3-D parameters that are not directly available from the image measurements. As for IBVS, we recall that the range of the object with respect to the camera appears in the coefficients of the interaction matrix related to the translational degrees of freedom. Noticeable exceptions are the schemes based on a numerical estima-

Other examples of temporal switching schemes can be found, such as the one developed in [24.48], to ensure the visibility of the target observed.

24.5.3 Feature Trajectory Planning

It is also possible to treat the optimization problem offline, during a planning stage. In this case, several constraints can be taken into account simultaneously, such as obstacle avoidance [24.49], joint limit and occlusions avoidance, and ensuring the visibility of the target [24.50]. The feature trajectories $s^*(t)$ that allow the camera to reach its desired pose while ensuring that the constraints are satisfied are determined using path planning techniques, such as the well-known potential field approach.

Coupling path planning with trajectory following also allows the robustness of the visual servo with respect to modeling errors to be significantly improved. Indeed, modeling errors may have large effects when the error $s - s^*$ is large, but have little effect when $s - s^*$ is small. Once the desired features trajectories $s^*(t)$ such that $s^*(0) = s(0)$ have been designed during the planning stage, it is easy to adapt the control scheme to take into account the fact that s^* is varying, and to make the error $s - s^*$ remain small. More precisely, we now have

$$\dot{e} = \dot{s} - \dot{s}^* = L_e v_c - \dot{s}^*,$$

from which we deduce, by selecting as usual $\dot{e} = -\lambda e$ as the desired behavior,

$$v_c = -\lambda \widehat{L}_e^+ e + \widehat{L}_e^+ \dot{s}^*.$$

The new second term of this control law anticipates the variation of s^* , removing the tracking error it would produce. We will see in the Sect. 24.7 that the form of the control law is similar when tracking of a moving target is considered.

tion of L_e or of L_e^+ (see Sect. 24.2.8). Another exception is the IBVS scheme that uses the constant matrix \widehat{L}_e^+ in the control scheme, in which only the depth for the desired pose is required, which is not so difficult to obtain in practice. As for PBVS and hybrid schemes that combine 2-D and 3-D data in e , 3-D parameters appear both in the error e to be regulated to 0 and in the interaction matrix.

A correct estimation of the 3-D parameters involved is thus important for IBVS since they will have an effect on the camera motion during the task execution (they appear in the stability conditions (24.13) and (24.14)), while a correct estimation is crucial in PBVS and hybrid schemes since they will have also an effect on the accuracy of the pose reached after convergence.

If a calibrated stereo vision system is used, all 3-D parameters can be easily determined by triangulation, as evoked in Sect. 24.2.5 and described in Chap. 23. Similarly, if a 3-D model of the object is known, all 3-D parameters can be computed from a pose-estimation algorithm. However, we recall that such an estimation can be quite unstable due to image noise. It is also possible to estimate 3-D parameters by using the epipolar geometry that relates the images of the same scene observed from different viewpoints. Indeed, in visual servoing, two images are generally available: the current one and the desired one.

Given a set of matches between the image measurements in the current image and in the desired one, the fundamental matrix, or the essential matrix if the camera is calibrated, can be recovered [24.6], and then used in visual servoing [24.51]. Indeed, from the essential matrix, the rotation and the translation up to a scalar factor between the two views can be estimated. However, near the convergence of the visual servo, that is, when the current and desired images are similar, the epipolar geometry becomes degenerate and it is not possible to estimate accurately the partial pose between the two views. For this reason, using homography is generally preferred.

Let \mathbf{x}_i and \mathbf{x}_i^* denote the homogeneous image coordinates for a point in the current and desired images. Then \mathbf{x}_i is related to \mathbf{x}_i^* by

$$\mathbf{x}_i = \mathbf{H}_i \mathbf{x}_i^*$$

in which \mathbf{H}_i is a homography matrix.

If all feature points lie on a 3-D plane, then there is a single homography matrix \mathbf{H} such that $\mathbf{x}_i = \mathbf{H} \mathbf{x}_i^*$ for all i . This homography can be estimated using the position of four matched points in the desired and the current images. If all the features points do not belong to the same 3-D plane, then three points can be used to define such a plane and five supplementary points are needed to estimate \mathbf{H} [24.52].

Once \mathbf{H} is available, it can be decomposed as

$$\mathbf{H} = \mathbf{R} + \frac{\mathbf{t}}{d^*} \mathbf{n}^{*\top}, \quad (24.27)$$

in which \mathbf{R} is the rotation matrix relating the orientation of the current and desired camera frames, \mathbf{n}^* is the normal to the chosen 3-D plane expressed in the desired frame, d^* is the distance to the 3-D plane from the desired frame, and \mathbf{t} is the translation between current and desired frames. From \mathbf{H} , it is thus possible to recover \mathbf{R} , \mathbf{t}/d^* , and \mathbf{n} . In fact, two solutions for these quantities exist [24.53], but it is quite easy to select the correct one using some knowledge about the desired pose. It is also possible to estimate the depth of any target point up to a common scale factor [24.50]. The unknown depth of each point that appears in classical IBVS can thus be expressed as a function of a single, constant parameter. Similarly, the pose parameters required by PBVS can be recovered up to a scalar factor as for the translation term. The PBVS schemes described previously can thus be revisited using this approach, with the new error defined as the translation up to a scalar factor and the angle/axis parameterization of the rotation. Finally, this approach has also been used for the hybrid visual servoing schemes described in Sect. 24.4.1. In that case, using such homography estimation, it has been possible to analyze the stability of hybrid visual servoing schemes in the presence of calibration errors [24.32].

24.7 Target Tracking

We now consider the case of a moving target and a constant desired value \mathbf{s}^* for the features, the generalization to varying desired features $\mathbf{s}^*(t)$ being immediate. The time variation of the error is now given by

$$\dot{\mathbf{e}} = \mathbf{L}_e \mathbf{v}_c + \frac{\partial \mathbf{e}}{\partial t}, \quad (24.28)$$

where the term $\partial \mathbf{e} / \partial t$ expresses the time variation of \mathbf{e} due to the generally unknown target motion. If the con-

trol law is still designed to try to ensure an exponential decoupled decrease of \mathbf{e} (that is, once again $\dot{\mathbf{e}} = -\lambda \mathbf{e}$), we now obtain using (24.28):

$$\mathbf{v}_c = -\lambda \widehat{\mathbf{L}}_e^+ \mathbf{e} - \widehat{\mathbf{L}}_e^+ \frac{\partial \mathbf{e}}{\partial t}, \quad (24.29)$$

where $\widehat{\partial \mathbf{e} / \partial t}$ is an estimation or an approximation of $\partial \mathbf{e} / \partial t$. This term must be introduced into the control law to compensate for the target motion.

Closing the loop, that is, inserting (24.29) into (24.28), we obtain

$$\dot{e} = -\lambda L_e \widehat{L}_e^+ e - L_e \widehat{L}_e^+ \frac{\partial e}{\partial t} + \frac{\partial e}{\partial t}. \quad (24.30)$$

Even if $L_e \widehat{L}_e^+ > 0$, the error will converge to zero only if the estimation of $\partial e / \partial t$ is sufficiently accurate so that

$$L_e \widehat{L}_e^+ \frac{\partial e}{\partial t} = \frac{\partial e}{\partial t}, \quad (24.31)$$

otherwise tracking errors will be observed. Indeed, by just solving the scalar differential equation $\dot{e} = -\lambda e + b$, which is a simplification of (24.30), we obtain $e(t) = e(0) \exp(-\lambda t) + b/\lambda$, which converges towards b/λ . On one hand, setting a high gain λ will reduce the tracking error, but on the other hand, setting the gain too high can make the system unstable. It is thus necessary to make b as small as possible.

Of course, if the system is known to be such that $\partial e / \partial t = 0$ (that is, the camera observes a motionless object, as described in Sect. 24.1), no tracking error will appear with the most simple estimation given by $\partial e / \partial t = 0$. Otherwise, a classical method in automatic control to cancel tracking errors consists of compensating the target motion through an integral term in the

control law. In this case, we have

$$\frac{\partial e}{\partial t} = \mu \sum_j e(j),$$

where μ is the integral gain that has to be tuned. This scheme allows the tracking errors to be canceled only if the target has a constant velocity. Other methods, based on feedforward control, estimate the term $\partial e / \partial t$ directly through the image measurements and the camera velocity, when it is available. Indeed, from (24.28), we obtain

$$\frac{\partial e}{\partial t} = \widehat{e} - \widehat{L}_e \widehat{v}_c,$$

where \widehat{e} can, for instance, be obtained as $\widehat{e}(t) = [e(t) - e(t - \Delta t)] / \Delta t$, Δt being the duration of the control loop. A Kalman filter [24.54] or more-elaborate filtering methods [24.55] can then be used to improve the estimated values obtained. If some knowledge about the target velocity or the target trajectory is available, it can of course be used to smooth or predict the motion [24.56–58]. For instance, in [24.59], the periodic motion of the heart and of the breath are compensated for an application of visual servoing in medical robotics. Finally, other methods have been developed to remove the perturbations induced by the target motion as fast as possible [24.39], using for instance predictive controllers [24.60].

24.8 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space

In the previous sections, we have considered the six components of the camera velocity as the input of the robot controller. As soon as the robot is not able to realize this motion, for instance, because it has fewer than six degrees of freedom, the control scheme must be expressed in the joint space. In this section, we describe how this can be done, and in the process develop a formulation for eye-to-hand systems.

In the joint space, the system equations for both the eye-to-hand and eye-in-hand configurations have the same form:

$$\dot{s} = J_s \dot{q} + \frac{\partial s}{\partial t}. \quad (24.32)$$

Here, $J_s \in \mathbb{R}^{k \times n}$ is the feature Jacobian matrix, which can be linked to the interaction matrix, and n is the number of robot joints.

For an eye-in-hand system (Fig. 24.15a), $\partial s / \partial t$ is the time variation of s due to a potential object motion, and

J_s is given by

$$J_s = L_s {}^c X_N J(q), \quad (24.33)$$

where

- ${}^c X_N$ is the spatial motion transform matrix (as defined in Chap. 1 and recalled in (24.15)) from the vision sensor frame to the end-effector frame. It is usually a constant matrix (as long as the vision sensor is rigidly attached to the end-effector). Thanks to the robustness of closed-loop control schemes, a coarse approximation of this transform matrix is sufficient in visual servoing. If needed, an accurate estimation is possible through classical hand-eye calibration methods [24.61].
- $J(q)$ is the robot Jacobian expressed in the end-effector frame (as defined in Chap. 1)

For an eye-to-hand system (Fig. 24.15b), $\partial s / \partial t$ is now the time variation of s due to a potential vision

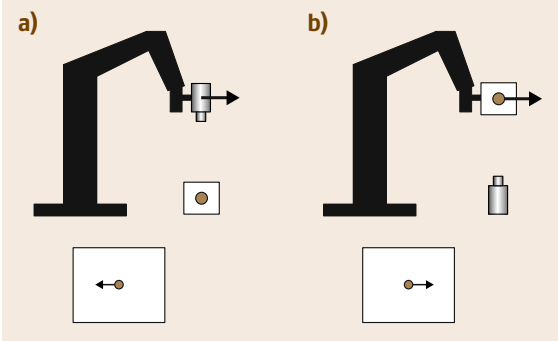


Fig. 24.15 (a) Eye-in-hand system, (b) eye-to-hand system: system schematic (top) and opposite image motion produced by the same robot motion (bottom)

sensor motion and \mathbf{J}_s can be expressed as:

$$\mathbf{J}_s = -\mathbf{L}_s^c \mathbf{X}_N^N \mathbf{J}(\mathbf{q}), \quad (24.34)$$

$$= -\mathbf{L}_s^c \mathbf{X}_0^0 \mathbf{J}(\mathbf{q}). \quad (24.35)$$

In (24.34), the classical robot Jacobian $^N \mathbf{J}(\mathbf{q})$ expressed in the end-effector frame is used but the spatial motion transform matrix $^c \mathbf{X}_N$ from the vision sensor frame to the end-effector frame changes all along the servo, and it has to be estimated at each iteration of the control scheme, usually using pose-estimation methods.

In (24.35), the robot Jacobian $^0 \mathbf{J}(\mathbf{q})$ is expressed in the robot reference frame, and the spatial motion transform matrix $^c \mathbf{X}_0$ from the vision sensor frame to that reference frame is constant as long as the camera does not move. In this case, which is convenient in practice, a coarse approximation of $^c \mathbf{X}_0$ is usually sufficient.

Once the modeling step is finished, it is quite easy to follow the procedure that has been used above to design a control scheme expressed in the joint space, and to determine the sufficient condition to ensure the stability of the control scheme. We obtain, considering

again $\mathbf{e} = \mathbf{s} - \mathbf{s}^*$, and an exponential decoupled decrease of \mathbf{e} :

$$\dot{\mathbf{q}} = -\lambda \widehat{\mathbf{J}_e^+} \mathbf{e} - \widehat{\mathbf{J}_e^+} \frac{\partial \mathbf{e}}{\partial t}. \quad (24.36)$$

If $k = n$, considering as in Sect. 24.1 the Lyapunov function $\mathcal{L} = \frac{1}{2} \|\mathbf{e}(t)\|^2$, a sufficient condition to ensure the global asymptotic stability is given by

$$\mathbf{J}_e \widehat{\mathbf{J}_e^+} > 0. \quad (24.37)$$

If $k > n$, we obtain similarly to Sect. 24.1

$$\widehat{\mathbf{J}_e^+} \mathbf{J}_e > 0 \quad (24.38)$$

to ensure the local asymptotic stability of the system. Note that the actual extrinsic camera parameters appear in \mathbf{J}_e while the estimated ones are used in $\widehat{\mathbf{J}_e^+}$. It is thus possible to analyze the robustness of the control scheme with respect to the camera extrinsic parameters. It is also possible to estimate directly the numerical value of \mathbf{J}_e or $\widehat{\mathbf{J}_e^+}$ using the methods described in Sect. 24.2.8.

Finally, to remove tracking errors, we have to ensure that

$$\mathbf{J}_e \widehat{\mathbf{J}_e^+} \frac{\partial \mathbf{e}}{\partial t} = \frac{\partial \mathbf{e}}{\partial t}.$$

Finally, let us note that, even if the robot has six degrees of freedom, it is generally not equivalent to first compute \mathbf{v}_c using (24.5) and then deduce $\dot{\mathbf{q}}$ using the robot inverse Jacobian, and to compute directly $\dot{\mathbf{q}}$ using (24.36). Indeed, it may occur that the robot Jacobian $\mathbf{J}(\mathbf{q})$ is singular while the feature Jacobian \mathbf{J}_s is not (that may occur when $k < n$). Furthermore, the properties of the pseudo-inverse ensure that using (24.5), $\|\mathbf{v}_c\|$ is minimal while using (24.36), $\|\dot{\mathbf{q}}\|$ is minimal. As soon as $\mathbf{J}_e^+ \neq \mathbf{J}^+(\mathbf{q})^N \mathbf{X}_c \mathbf{L}_e^+$, the control schemes will be different and will induce different robot trajectories. The choice of the state space is thus important.

24.9 Conclusions

In this chapter we have only considered velocity controllers, which are convenient for most classical robot arms. However, the dynamics of the robot must of course be taken into account for high-speed tasks, or when we deal with mobile nonholonomic or underactuated robots. As for sensors, we have only considered geometrical features coming from a classical perspective camera. Features related to the image motion or coming from other vision sensors (fish-eye camera, catadioptric camera, echographic probes, etc.) necessitate reconsideration of the modeling issues to select adequate visual features. Finally, fusing visual features with data coming from other sensors (force sensor, proximity sensors, etc.) at the level of the control scheme will allow new research topics to be addressed. Numerous ambitious applications of visual servoing can also be considered, including mobile robots in indoor and outdoor environments, for aerial, space, and submarine robots, and in medical robotics. The end of fruitful research in the field of visual servo is thus nowhere yet in sight.

References

- 24.1 B. Espiau, F. Chaumette, P. Rives: A new approach to visual servoing in robotics, *IEEE Trans. Robot. Autom.* **8**, 313–326 (1992)
- 24.2 S. Hutchinson, G. Hager, P. Corke: A tutorial on visual servo control, *IEEE Trans. Robot. Autom.* **12**, 651–670 (1996)
- 24.3 L. Weiss, A. Sanderson, C. Neuman: Dynamic sensor-based control of robots with visual feedback, *IEEE J. Robot. Autom.* **3**, 404–417 (1987)
- 24.4 J. Feddema, O. Mitchell: Vision-guided servoing with feature-based trajectory generation, *IEEE Trans. Robot. Autom.* **5**, 691–700 (1989)
- 24.5 D. Forsyth, J. Ponce: *Computer Vision: A Modern Approach* (Prentice Hall, Upper Saddle River 2003)
- 24.6 Y. Ma, S. Soatto, J. Kosecka, S. Sastry: *An Invitation to 3-D Vision: From Images to Geometric Models* (Springer, New York 2003)
- 24.7 H. Michel, P. Rives: Singularities in the determination of the situation of a robot effector from the erspective view of three points. Tech. Rep. 1850, INRIA Res. Rep. (1993)
- 24.8 M. Fischler, R. Bolles: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM* **24**, 381–395 (1981)
- 24.9 E. Malis: Improving vision-based control using efficient second-order minimization techniques, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp.1843–1848
- 24.10 P. Corke, S. Hutchinson: A new partitioned approach to image-based visual servo control, *IEEE Trans. Robot. Autom.* **17**, 507–515 (2001)
- 24.11 F. Chaumette: Potential problems of stability and convergence in image-based and position-based visual servoing. In: *The Confluence of Vision and Control*, LNCIS Series, Vol. 237, ed. by D. Kriegman, G. Hager, S. Morse (Springer, Heidelberg 1998) pp. 66–78
- 24.12 E. Malis: Visual servoing invariant to changes in camera intrinsic parameters, *IEEE Trans. Robot. Autom.* **20**, 72–81 (2004)
- 24.13 A. Isidori: *Nonlinear Control Systems*, 3rd edn. (Springer Berlin, Heidelberg 1995)
- 24.14 G. Hager, W. Chang, A. Morse: Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination, *IEEE Contr. Syst. Mag.* **15**, 30–39 (1995)
- 24.15 M. Iwatsuki, N. Okiyama: A new formulation of visual servoing based on cylindrical coordinate system, *IEEE Trans. Robot. Automation* **21**, 266–273 (2005)
- 24.16 F. Chaumette, P. Rives, B. Espiau: Classification and realization of the different vision-based tasks. In: *Visual Servoing*, Robotics and Automated Systems, Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 199–228
- 24.17 A. Castano, S. Hutchinson: Visual compliance: task directed visual servo control, *IEEE Trans. Robot. Autom.* **10**, 334–342 (1994)
- 24.18 G. Hager: A modular system for robust positioning using feedback from stereo vision, *IEEE Trans. Robot. Autom.* **13**, 582–595 (1997)
- 24.19 F. Chaumette: Image moments: a general and useful set of features for visual servoing, *IEEE Trans. Robot. Autom.* **20**, 713–723 (2004)
- 24.20 O. Tahri, F. Chaumette: Point-based and region-based image moments for visual servoing of planar objects, *IEEE Trans. Robot.* **21**, 1116–1127 (2005)
- 24.21 I. Suh: Visual servoing of robot manipulators by fuzzy membership function based neural networks. In: *Visual Servoing*, Robotics and Automated Systems, Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 285–315
- 24.22 G. Wells, C. Venaille, C. Torras: Vision-based robot positioning using neural networks, *Image Vision Comput.* **14**, 75–732 (1996)
- 24.23 J.T. Lapresté, F. Jurie, F. Chaumette: An efficient method to compute the inverse jacobian matrix in visual servoing, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp. 727–732
- 24.24 K. Hosada, M. Asada: Versatile visual servoing without knowledge of true jacobian, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Munchen 1994) pp. 186–193
- 24.25 M. Jägersand, O. Fuentes, R. Nelson: Experimental evaluation of uncalibrated visual servoing for precision manipulation, *IEEE Int. Conf. Robot. Autom.* (Albuquerque 1997) pp. 2874–2880
- 24.26 J. Piepmeyer, G.M. Murray, H. Lipkin: Uncalibrated dynamic visual servoing, *IEEE Trans. Robot. Autom.* **20**, 143–147 (2004)
- 24.27 K. Deguchi: Direct interpretation of dynamic images and camera motion for visual servoing without image feature correspondence, *J. Robot. Mechatron.* **9**(2), 104–110 (1997)
- 24.28 W. Wilson, C. Hulls, G. Bell: Relative end-effector control using cartesian position based visual servoing, *IEEE Trans. Robot. Autom.* **12**, 684–696 (1996)
- 24.29 B. Thuilot, P. Martinet, L. Cordesses, J. Gallice: Position based visual servoing: Keeping the object in the field of vision, *IEEE Int. Conf. Robot. Autom.* (Washington 2002) pp. 1624–1629
- 24.30 D. Dementhon, L. Davis: Model-based object pose in 25 lines of code, *Int. J. Comput. Vision* **15**, 123–141 (1995)

- 24.31 D. Lowe: Three-dimensional object recognition from single two-dimensional images, *Artif. Intell.* **31**(3), 355–395 (1987)
- 24.32 E. Malis, F. Chaumette, S. Boudet: 2-1/2 D visual servoing, *IEEE Trans. Robot. Autom.* **15**, 238–250 (1999)
- 24.33 E. Malis, F. Chaumette: Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods, *IEEE Trans. Robot. Autom.* **18**, 176–186 (2002)
- 24.34 J. Chen, D. Dawson, W. Dixon, A. Behal: Adaptive homography-based visual servo tracking for fixed camera-in-hand configurations, *IEEE Trans. Contr. Syst. Technol.* **13**, 814–825 (2005)
- 24.35 G. Morel, T. Leibzeit, J. Szewczyk, S. Boudet, J. Pot: Explicit incorporation of 2-D constraints in vision-based control of robot manipulators, *Int. Symp. Exp. Robot.* **250**, 99–108 (2000), LNCIS Series
- 24.36 F. Chaumette, E. Malis: 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings, *IEEE Int. Conf. Robot. Autom.* (San Fransisco 2000) pp. 630–635
- 24.37 E. Cervera, A.D. Pobil, F. Berry, P. Martinet: Improving image-based visual servoing with three-dimensional features, *Int. J. Robot. Res.* **22**, 821–840 (2004)
- 24.38 F. Schramm, G. Morel, A. Micaelli, A. Lottin: Extended 2-D visual servoing, *IEEE Int. Conf. Robot. Autom.* (New Orleans 2004) pp. 267–273
- 24.39 N. Papanikolopoulos, P. Khosla, T. Kanade: Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control, *IEEE Trans. Robot. Autom.* **9**, 14–35 (1993)
- 24.40 K. Hashimoto, H. Kimura: LQ optimal and nonlinear approaches to visual servoing. In: *Visual Servoing*, *Robot. Autom. Syst.*, Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 165–198
- 24.41 B. Nelson, P. Khosla: Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance, *Int. J. Robot. Res.* **14**, 225–269 (1995)
- 24.42 B. Nelson, P. Khosla: Force and vision resolvability for assimilating disparate sensory feedback, *IEEE Trans. Robot. Autom.* **12**, 714–731 (1996)
- 24.43 R. Sharma, S. Hutchinson: Motion perceptibility and its application to active vision-based servo control, *IEEE Trans. Robot. Autom.* **13**, 607–617 (1997)
- 24.44 E. Marchand, F. Chaumette, A. Rizzo: Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Osaka 1996) pp. 1083–1090
- 24.45 E. Marchand, G. Hager: Dynamic sensor planning in visual servoing, *IEEE Int. Conf. Robot. Autom.* (Leuven) (1998) pp. 1988–1993
- 24.46 N. Cowan, J. Weingarten, D. Koditschek: Visual servoing via navigation functions, *IEEE Trans. Robot. Autom.* **18**, 521–533 (2002)
- 24.47 N. Gans, S. Hutchinson: An asymptotically stable switched system visual controller for eye in hand robots, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Las Vegas 2003) pp. 735–742
- 24.48 G. Chesi, K. Hashimoto, D. Prattichizio, A. Vicino: Keeping features in the field of view in eye-in-hand visual servoing: a switching approach, *IEEE Trans. Robot. Autom.* **20**, 908–913 (2004)
- 24.49 K. Hosoda, K. Sakamoto, M. Asada: Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3-D reconstruction, *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Vol. 3 (Pittsburgh 1995) pp. 29–34
- 24.50 Y. Mezouar, F. Chaumette: Path planning for robust image-based control, *IEEE Trans. Robot. Autom.* **18**, 534–549 (2002)
- 24.51 R. Basri, E. Rivlin, I. Shimshoni: Visual homing: Surfing on the epipoles, *Int. J. Comput. Vision* **33**, 117–137 (1999)
- 24.52 E. Malis, F. Chaumette, S. Boudet: 2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement, *Int. J. Comput. Vision* **37**, 79–97 (2000)
- 24.53 O. Faugeras: *Three-Dimensional Computer Vision: a Geometric Viewpoint* (MIT Press, Cambridge 1993)
- 24.54 P. Corke, M. Goods: Controller design for high performance visual servoing, 12th World Congress IFAC'93 (Sydney 1993) pp. 395–398
- 24.55 F. Bensalah, F. Chaumette: Compensation of abrupt motion changes in target tracking by visual servoing, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Pittsburgh 1995) pp. 181–187
- 24.56 P. Allen, B. Yoshimi, A. Timcenko, P. Michelman: Automated tracking and grasping of a moving object with a robotic hand-eye system, *IEEE Trans. Robot. Autom.* **9**, 152–165 (1993)
- 24.57 K. Hashimoto, H. Kimura: Visual servoing with non linear observer, *IEEE Int. Conf. Robot. Autom.* (Nagoya 1995) pp. 484–489
- 24.58 A. Rizzi, D. Koditschek: An active visual estimator for dexterous manipulation, *IEEE Trans. Robot. Autom.* **12**, 697–713 (1996)
- 24.59 R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M.A. Sanchez, J. Marescaux: Active filtering of physiological motion in robotized surgery using predictive control, *IEEE Trans. Robot.* **21**, 67–79 (2005)
- 24.60 J. Gangloff, M. de Mathelin: Visual servoing of a 6 dof manipulator for unknown 3-D profile following, *IEEE Trans. Robot. Autom.* **18**, 511–520 (2002)
- 24.61 R. Tsai, R. Lenz: A new technique for fully autonomous efficient 3-D robotics hand-eye calibration, *IEEE Trans. Robot. Autom.* **5**, 345–358 (1989)

Multisensor

25. Multisensor Data Fusion

Hugh Durrant-Whyte, Thomas C. Henderson

Multisensor data fusion is the process of combining observations from a number of different sensors to provide a robust and complete description of an environment or process of interest. Data fusion finds wide application in many areas of robotics such as object recognition, environment mapping, and localization.

This chapter has three parts: methods, architectures, and applications. Most current data fusion methods employ probabilistic descriptions of observations and processes and use Bayes' rule to combine this information. This chapter surveys the main probabilistic modeling and fusion techniques including grid-based models, Kalman filtering, and sequential Monte Carlo techniques. This chapter also briefly reviews a number of non-probabilistic data fusion methods. Data fusion systems are often complex combinations of sensor devices, processing, and fusion algorithms. This chapter provides an overview of key principles in data fusion architectures from both a hardware and algorithmic viewpoint. The applications of data fusion are pervasive in robotics and underly the core problem of sensing, estimation, and perception. We highlight two example applications that bring out these features. The first describes a navigation or self-tracking application for an autonomous vehicle. The second describes an

- 25.1 **Multisensor Data Fusion Methods**..... 585
 - 25.1.1 Bayes' Rule 586
 - 25.1.2 Probabilistic Grids 589
 - 25.1.3 The Kalman Filter 590
 - 25.1.4 Sequential Monte Carlo Methods 594
 - 25.1.5 Alternatives to Probability 596
- 25.2 **Multisensor Fusion Architectures**..... 598
 - 25.2.1 Architectural Taxonomy 598
 - 25.2.2 Centralized, Local Interaction, and Hierarchical 600
 - 25.2.3 Decentralized, Global Interaction, and Heterarchical 600
 - 25.2.4 Decentralized, Local Interaction, and Hierarchical 601
 - 25.2.5 Decentralized, Local Interaction, and Heterarchical 602
- 25.3 **Applications**..... 603
 - 25.3.1 Dynamic System Control 603
 - 25.3.2 ANSER II: Decentralized Data Fusion 604
- 25.4 **Conclusions**..... 608
- References** 608

application in mapping and environment modeling.

The essential algorithmic tools of data fusion are reasonably well established. However, the development and use of these tools in realistic robotics applications is still developing.

25.1 Multisensor Data Fusion Methods

The most widely used data fusion methods employed in robotics originate in the fields of statistics, estimation, and control. However, the application of these methods in robotics has a number of unique features and challenges. In particular, autonomy is most often the goal and so results must be presented and interpreted in

a form from which autonomous decisions can be made, for recognition or navigation, for example.

In this section we review the main data fusion methods employed in robotics. These are very often based on probabilistic methods, which are indeed now considered the standard approach to data fusion in all robotics

applications [25.1]. Probabilistic data fusion methods are generally based on Bayes' rule for combining prior and observation information. Practically, this may be implemented in a number of ways: through the use of the Kalman and extended Kalman filters, through sequential Monte Carlo methods, or through the use of functional density estimates. Each of these is reviewed. There are a number of alternatives to probabilistic methods. These include the theory of evidence and interval methods. Such alternative techniques are not as widely used as they once were, however they have some special features that can be advantageous in specific problems. These, too, are briefly reviewed.

25.1.1 Bayes' Rule

Bayes' rule lies at the heart of most data fusion methods. In general, Bayes' rule provides a means to make inferences about an object or environment of interest described by a state \mathbf{x} , given an observation \mathbf{z} .

Bayesian Inference

Bayes' rule requires that the relationship between \mathbf{x} and \mathbf{z} be encoded as a joint probability or joint probability distribution $P(\mathbf{x}, \mathbf{z})$ for discrete and continuous variables, respectively. The chain rule of conditional probabilities can be used to expand a joint probability in two ways

$$P(\mathbf{x}, \mathbf{z}) = P(\mathbf{x} | \mathbf{z})P(\mathbf{z}) = P(\mathbf{z} | \mathbf{x})P(\mathbf{x}). \quad (25.1)$$

Rearranging in terms of one of the conditionals, Bayes' rule is obtained

$$P(\mathbf{x} | \mathbf{z}) = \frac{P(\mathbf{z} | \mathbf{x})P(\mathbf{x})}{P(\mathbf{z})}. \quad (25.2)$$

The value of this result lies in the interpretation of the probabilities $P(\mathbf{x} | \mathbf{z})$, $P(\mathbf{z} | \mathbf{x})$, and $P(\mathbf{x})$. Suppose it is necessary to determine the various likelihoods of different values of an unknown state \mathbf{x} . There may be prior beliefs about what values of \mathbf{x} might be expected, encoded in the form of relative likelihoods in the *prior probability* $P(\mathbf{x})$. To obtain more information about the state \mathbf{x} an observation \mathbf{z} is made. These observations are modeled in the form of a conditional probability $P(\mathbf{z} | \mathbf{x})$ that describes, for each fixed state \mathbf{x} , the probability that the observation \mathbf{z} will be made, i. e., the probability of \mathbf{z} given \mathbf{x} . The new likelihoods associated with the state \mathbf{x} are computed from the product of the original prior information and the information gained by observation. This is encoded in the *posterior probability* $P(\mathbf{x} | \mathbf{z})$, which describes the likelihoods associated with \mathbf{x} given the observation \mathbf{z} . In this fusion process, the marginal probability $P(\mathbf{z})$ simply serves to normalize the posterior and

is not generally computed. The marginal $P(\mathbf{z})$ plays an important role in model validation or data association as it provides a measure of how well the observation is predicted by the prior, because $P(\mathbf{z}) = \int P(\mathbf{z} | \mathbf{x})P(\mathbf{x})d\mathbf{x}$. The value of Bayes' rule is that it provides a principled means of combining observed information with prior beliefs about the state of the world.

Sensor Models and Multisensor Bayesian Inference

The conditional probability $P(\mathbf{z} | \mathbf{x})$ serves the role of a *sensor model* and can be thought of in two ways. First, in building a sensor model, the probability is constructed by fixing the value of $\mathbf{x} = \mathbf{x}$ and then asking what probability density $P(\mathbf{z} | \mathbf{x} = \mathbf{x})$ on \mathbf{z} results. Conversely, when this sensor model is used and observations are made, $\mathbf{z} = \mathbf{z}$ is fixed and a *likelihood function* $P(\mathbf{z} = \mathbf{z} | \mathbf{x})$ on \mathbf{x} is inferred. The likelihood function, while not strictly a probability density, models the relative likelihood that different values of \mathbf{x} gave rise to the observed value of \mathbf{z} . The product of this likelihood with the prior, both defined on \mathbf{x} , gives the posterior or observation update $P(\mathbf{x} | \mathbf{z})$. In a practical implementation of (25.2), $P(\mathbf{z} | \mathbf{x})$ is constructed as a function of both variables (or a matrix in discrete form). For each fixed value of \mathbf{x} , a probability density on \mathbf{z} is defined. Therefore, as \mathbf{x} varies, a family of likelihoods on \mathbf{z} is created.

The multisensor form of Bayes' rule requires conditional independence

$$\begin{aligned} P(\mathbf{z}_1, \dots, \mathbf{z}_n | \mathbf{x}) &= P(\mathbf{z}_1 | \mathbf{x}) \cdots P(\mathbf{z}_n | \mathbf{x}) \\ &= \prod_{i=1}^n P(\mathbf{z}_i | \mathbf{x}). \end{aligned} \quad (25.3)$$

so that

$$P(\mathbf{x} | \mathbf{Z}^n) = CP(\mathbf{x}) \prod_{i=1}^n P(\mathbf{z}_i | \mathbf{x}), \quad (25.4)$$

where C is a normalizing constant. Equation (25.4) is known as the *independent likelihood pool* [25.2]. This states that the posterior probability on \mathbf{x} given all observations \mathbf{Z}^n , is simply proportional to the product of prior probability and individual likelihoods from each information source.

The recursive form of Bayes' rule is

$$P(\mathbf{x} | \mathbf{Z}^k) = \frac{P(\mathbf{z}_k | \mathbf{x})P(\mathbf{x} | \mathbf{Z}^{k-1})}{P(\mathbf{z}_k | \mathbf{Z}^{k-1})}. \quad (25.5)$$

The advantage of (25.5) is that we need to compute and store only the posterior density $P(\mathbf{x} | \mathbf{Z}^{k-1})$, which contains a complete summary of all past information.

When the next piece of information $P(z_k | \mathbf{x})$ arrives, the previous posterior takes on the role of the current prior and the product of the two becomes, when normalized, the new posterior.

Bayesian Filtering

Filtering is concerned with the sequential process of maintaining a probabilistic model for a state which evolves over time and which is periodically observed

by a sensor. Filtering forms the basis for many problems in tracking and navigation. The general filtering problem can be formulated in Bayesian form. This is significant because it provides a common representation for a range of discrete and continuous data fusion problems without recourse to specific target or observation models.

Define \mathbf{x}_t as the value of a state of interest at time t . This may, for example, describe a feature to be tracked, the state of a process being monitored, or the location

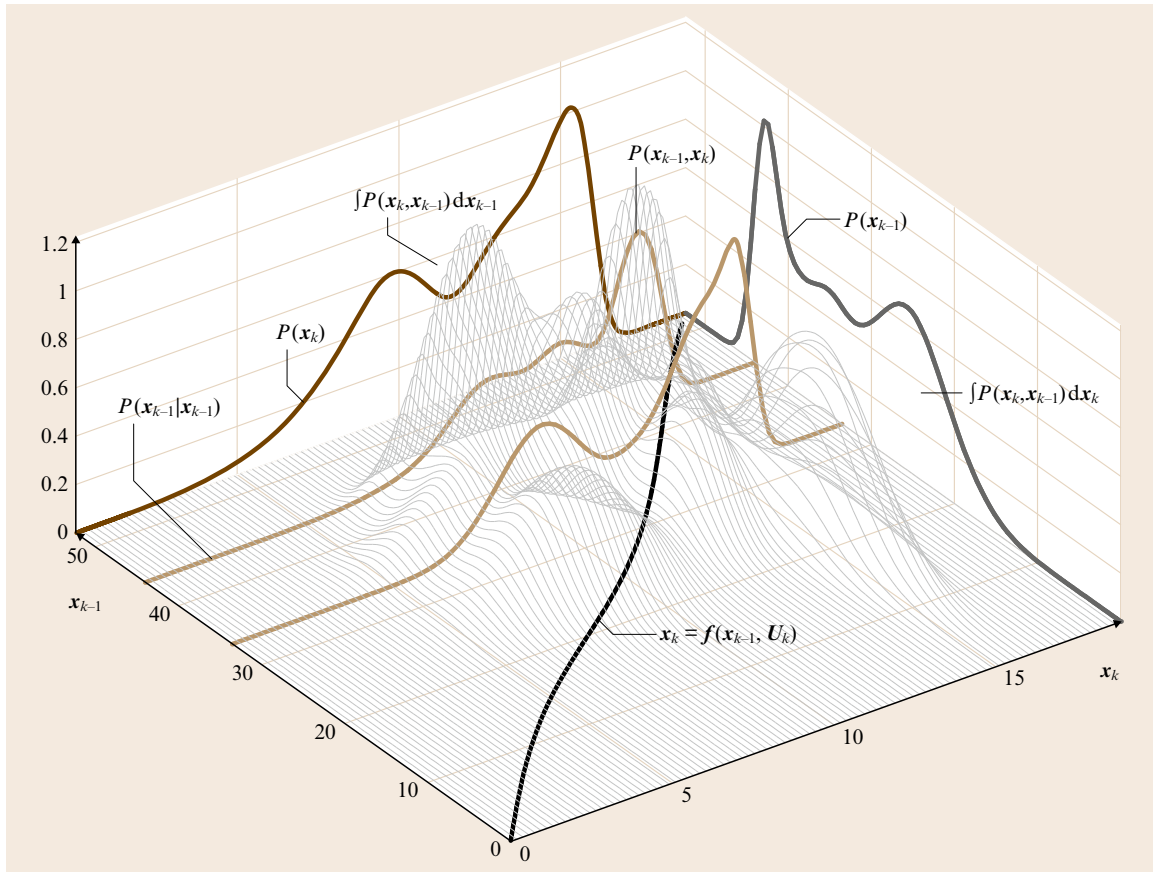


Fig. 25.1 Time update step for the full Bayes filter. At a time $k-1$, knowledge of the state \mathbf{x}_{k-1} is summarized in a probability distribution $P(\mathbf{x}_{k-1})$. A vehicle model, in the form of a conditional probability density $P(\mathbf{x}_k | \mathbf{x}_{k-1})$, then describes the stochastic transition of the vehicle from a state \mathbf{x}_{k-1} at a time $k-1$ to a state \mathbf{x}_k at a time k . Functionally, this state transition may be related to an underlying kinematic state model in the form $\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k)$. The figure shows two typical conditional probability distributions $P(\mathbf{x}_k | \mathbf{x}_{k-1})$ on the state \mathbf{x}_k given fixed values of \mathbf{x}_{k-1} . The product of this conditional distribution with the marginal distribution $P(\mathbf{x}_{k-1})$, describing the prior likelihood of values of \mathbf{x}_k , gives the joint distribution $P(\mathbf{x}_k, \mathbf{x}_{k-1})$ shown as the surface in the figure. The total marginal density $P(\mathbf{x}_k)$ describes knowledge of \mathbf{x}_k after state transition has occurred. The marginal density $P(\mathbf{x}_k)$ is obtained by integrating (projecting) the joint distribution $P(\mathbf{x}_k, \mathbf{x}_{k-1})$ over all \mathbf{x}_{k-1} . Equivalently, using the total probability theorem, the marginal density can be obtained by integrating (summing) all conditional densities $P(\mathbf{x}_k | \mathbf{x}_{k-1})$ weighted by the prior probability $P(\mathbf{x}_{k-1})$ of each \mathbf{x}_{k-1} . The process can equally be run in reverse (a retroverse motion model) to obtain $P(\mathbf{x}_{k-1})$ from $P(\mathbf{x}_k)$ given a model $P(\mathbf{x}_{k-1} | \mathbf{x}_k)$.

of a platform for which navigation data is required. For convenience, and without loss of generality, time is defined at discrete (asynchronous) times $t_k \triangleq k$. At a time instant k , the following quantities are defined:

\mathbf{x}_k : the state vector to be estimated at time k
 \mathbf{u}_k : a control vector, assumed known, and applied at time $k - 1$ to drive the state from \mathbf{x}_{k-1} to \mathbf{x}_k at time k
 z_k : an observation taken of the state \mathbf{x}_k at time k

In addition, the following sets are also defined.

- The history of states:
 $\mathbf{X}^k = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_k\} = \{\mathbf{X}^{k-1}, \mathbf{x}_k\}$.
- The history of control inputs:
 $\mathbf{U}^k = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k\} = \{\mathbf{U}^{k-1}, \mathbf{u}_k\}$.
- The history of state observations:
 $\mathbf{Z}^k = \{z_1, z_2, \dots, z_k\} = \{\mathbf{Z}^{k-1}, z_k\}$.

In probabilistic form, the general data fusion problem is to find the posterior density

$$P(\mathbf{x}_k | \mathbf{Z}^k, \mathbf{U}^k, \mathbf{x}_0) \quad (25.6)$$

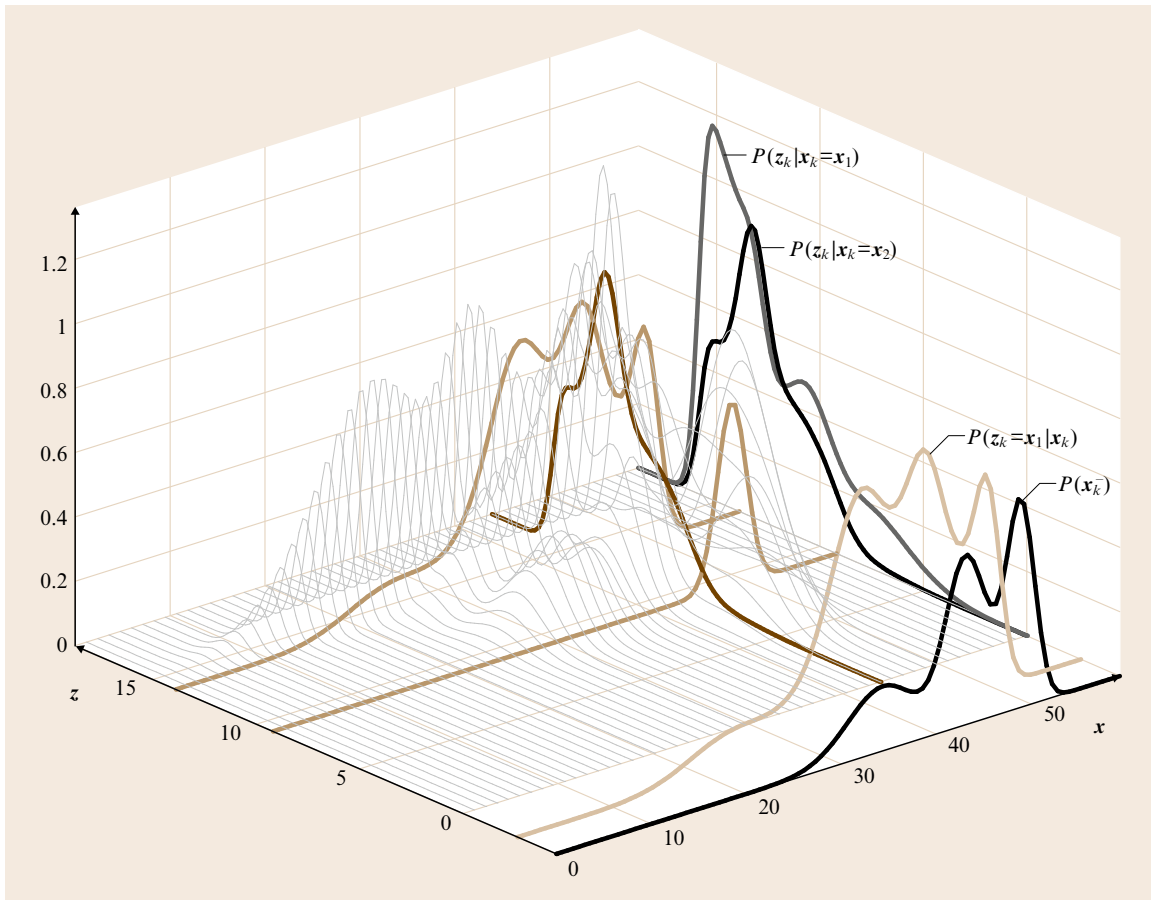


Fig. 25.2 Observation update for the full Bayes filter. Prior to observation, an observation model in the form of the conditional density $P(z_k | \mathbf{x}_k)$ is established. For a fixed value of \mathbf{x}_k , equal to \mathbf{x}_1 or \mathbf{x}_2 for example, a density function $P(z_k | \mathbf{x}_k = \mathbf{x}_1)$ or $P(z_k | \mathbf{x}_k = \mathbf{x}_2)$ is defined describing the likelihood of making the observation z_k . Together the density $P(z_k | \mathbf{x}_k)$ is then a function of both z_k and \mathbf{x}_k . This conditional density then defines the observation model. Now, in operation, a specific observation $z_k = \mathbf{x}_1$ is made and the resulting distribution $P(z_k = \mathbf{x}_1 | \mathbf{x}_k)$ defines a density function (now termed the likelihood function) on \mathbf{x}_k . This density is then multiplied by the prior density $P(\mathbf{x}_k^-)$ and normalized to obtain the posterior distribution $P(\mathbf{x}_k | z_k)$ describing knowledge in the state after observation

for all times k given the recorded observations and control inputs up to and including time k together (possibly) with knowledge of the initial state \mathbf{x}_0 . Bayes' rule can be used to write (25.6) in terms of a sensor model $P(\mathbf{z}_k | \mathbf{x}_k)$ and a predicted probability density $P(\mathbf{x}_k | \mathbf{Z}^{k-1}, \mathbf{U}^k, \mathbf{x}_0)$ based on observations up to time $k-1$ as

$$\begin{aligned} P(\mathbf{x}_k | \mathbf{Z}^k, \mathbf{U}^k, \mathbf{x}_0) \\ = \frac{P(\mathbf{z}_k | \mathbf{x}_k)P(\mathbf{x}_k | \mathbf{Z}^{k-1}, \mathbf{U}^k, \mathbf{x}_0)}{P(\mathbf{z}_k | \mathbf{Z}^{k-1}, \mathbf{U}^k)}. \end{aligned} \quad (25.7)$$

The denominator in (25.7) is independent of the state and following (25.4) can be set to some normalizing constant C . The sensor model makes use of the conditional independence assumption from (25.3).

The total probability theorem can be used to rewrite the second term in the numerator of (25.7) in terms of the state transition model and the joint posterior from time step $k-1$ as

$$\begin{aligned} P(\mathbf{x}_k | \mathbf{Z}^{k-1}, \mathbf{U}^k, \mathbf{x}_0) \\ = \int P(\mathbf{x}_k, \mathbf{x}_{k-1} | \mathbf{Z}^{k-1}, \mathbf{U}^k, \mathbf{x}_0) d\mathbf{x}_{k-1} \\ = \int P(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{Z}^{k-1}, \mathbf{U}^k, \mathbf{x}_0) \\ \times P(\mathbf{x}_{k-1} | \mathbf{Z}^{k-1}, \mathbf{U}^k, \mathbf{x}_0) d\mathbf{x}_{k-1} \\ = \int P(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_k) \\ \times P(\mathbf{x}_{k-1} | \mathbf{Z}^{k-1}, \mathbf{U}^{k-1}, \mathbf{x}_0) d\mathbf{x}_{k-1}, \end{aligned} \quad (25.8)$$

where the last equality implies that the future state depends only on the current state and the control exerted at this time. The state transition model is described in terms of a probability distribution in the form $P(\mathbf{x}_k | \mathbf{x}_{k-1}, \mathbf{u}_k)$. That is, the state transition may reasonably be assumed to be a Markov process in which the next state \mathbf{x}_k depends only on the immediately preceding state \mathbf{x}_{k-1} and the applied control \mathbf{u}_k , and is independent of both the observations and preceding states.

Equations (25.7) and (25.8) define a recursive solution to (25.6). Equation (25.8) is the *time update* or prediction step for the full Bayes data fusion algorithm. A graphical description of this equation is shown in Fig. 25.1. Equation (25.7) is the *observation update* step for the full Bayes data fusion algorithm. A graphical description of this equation is shown in Fig. 25.2. The Kalman filter, grid-based methods, and sequential Monte Carlo methods, to be described, are specific implementations of these general equations.

25.1.2 Probabilistic Grids

Probabilistic grids are conceptually the simplest way of implementing Bayesian data fusion methods. They can be applied both to problems in mapping [25.3, 4] and tracking [25.5].

In mapping applications, the environment of interest is divided into a grid of equally sized spatial cells. Each cell is indexed and labeled with a property, thus the state \mathbf{x}_{ij} may describe a two-dimensional world indexed by ij and having the property x . Interest is focused on maintaining a probability distribution on possible state values $P(\mathbf{x}_{ij})$ at each grid cell. Typically, in navigation and mapping problems, the property of interest has only two values O and E , *occupied* and *empty*, respectively, and it is then usual to assume that $P(\mathbf{x}_{ij} = O) = 1 - P(\mathbf{x}_{ij} = E)$. However, there is no particular constraint on the property encoded by the state \mathbf{x}_{ij} which could have many values (green, red, blue, for example) and indeed be continuous (the temperature at a cell, for example).

Once the state has been defined, Bayesian methods require that a sensor model or likelihood function for the sensor be established. In theory, this requires specification of a probability distribution $P(\mathbf{z} | \mathbf{x}_{ij} = x_{ij})$ mapping each possible grid state \mathbf{x}_{ij} to a distribution on observations. Practically, however, this is implemented simply as another *observation* grid so that for a specific observation $\mathbf{z} = z$ (taken from a specific location), a grid of likelihoods on the states \mathbf{x}_{ij} is produced in the form $P(\mathbf{z} = z | \mathbf{x}_{ij}) = \Lambda(\mathbf{x}_{ij})$. It is then trivial to apply Bayes' rule to update the property value at each grid cell as

$$P^+(\mathbf{x}_{ij}) = C \Lambda(\mathbf{x}_{ij}) P(\mathbf{x}_{ij}), \quad \forall i, j, \quad (25.9)$$

where C is a normalizing constant obtained by summing posterior probabilities to one at node ij only. Computationally, this is a simple pointwise multiplication of two grids. Some care needs to be taken that the two grids appropriately overlap and align with each other at the right scale. In some instances it is also valuable to encode the fact that spatially adjacent cells will influence each other; that is, if we knew the value of the property (occupancy, temperature, for example) at ij we will have some belief also of the value of this property at adjacent nodes $i+1, j, i, j+1$, etc. Different sensors and the fusion of different sensor outputs is accommodated simply by building appropriate sensor models $\Lambda(\mathbf{x}_{ij})$.

Grids can also be used for tracking and self-tracking (localization). The state \mathbf{x}_{ij} in this case is the location of the entity being tracked. This is a qualitatively different definition of state from that used in mapping. The probability $P(\mathbf{x}_{ij})$ must now be interpreted as the probability

that the object being tracked occupies the grid cell ij . In the case of mapping, the sum of property probabilities at each grid cell is one, whereas in the case of tracking, the sum of location probabilities over *the whole grid* must sum to one. Otherwise, the procedure for updating is very similar. An observation grid is constructed which, when instantiated with an observation value, provides a location likelihood grid $P(z = z | x_{ij}) = \Lambda(x_{ij})$. Bayes' rule is then applied to update the location probability at each grid cell in the same form as (25.9) except that now the normalization constant C is obtained by summing posterior probabilities over *all* ij grid cells. This can become computationally expensive, especially if the grid has three or more dimensions. One major advantage of grid-based tracking is that it is easy to incorporate quite complex prior information. For example, if it is known that the object being tracked is on a road, then the probability location values for all off-road grid cells can simply be set to zero.

Grid-based fusion is appropriate to situations where the domain size and dimension are modest. In such cases, grid-based methods provide straightforward and effective fusion algorithms. Grid-based methods can be extended in a number of ways; to hierarchical (quad-tree) grids, or to irregular (triangular, pentagonal) grids. These can help reduce computation in larger spaces. Monte Carlo and particle filtering methods (Sect. 25.1.4) may be considered as grid-based methods, where the grid cells themselves are sample of the underlying probability density for the state.

25.1.3 The Kalman Filter

The Kalman filter is a recursive linear estimator that successively calculates an estimate for a continuous valued state, that evolves over time, on the basis of periodic observations of the state. The Kalman filter employs an explicit statistical model of how the parameter of interest $x(t)$ evolves over time and an explicit statistical model of how the observations $z(t)$ that are made are related to this parameter. The gains employed in a Kalman filter are chosen to ensure that, with certain assumptions about the observation and process models used, the resulting estimate $\hat{x}(t)$ minimizes mean-squared error and is thus the conditional mean $\hat{x}(t) = E[x(t) | Z^t]$: an average, rather than a most likely value.

The Kalman filter has a number of features which make it ideally suited to dealing with complex multisensor estimation and data fusion problems. In particular, the explicit description of process and observations allows a wide variety of different sensor models to be

incorporated within the basic algorithm. In addition, the consistent use of statistical measures of uncertainty makes it possible to quantitatively evaluate the role each sensor plays in overall system performance. Further, the linear recursive nature of the algorithm ensures that its application is simple and efficient. For these reasons, the Kalman filter has found widespread application in many different data fusion problems [25.6–9].

In robotics, the Kalman filter is most suited to problems in tracking, localization, and navigation, and less so to problems in mapping. This is because the algorithm works best with well-defined state descriptions (positions, velocities, for example), and for states where observation and time-propagation models are also well understood.

Observation and Transition Models

The Kalman filter may be considered a specific instance of the recursive Bayesian filter of (25.7, 25.8) for the case where the probability densities on states are Gaussian. The starting point for the Kalman Filter algorithm is to define a model for the states to be estimated in the standard state-space form:

$$\dot{x}(t) = F(t)x(t) + B(t)u(t) + G(t)v(t), \quad (25.10)$$

where $x(t)$ is the state vector of interest, $u(t)$ is a known control input, $v(t)$ is a random variable describing uncertainty in the evolution of the state, and $F(t)$, $B(t)$, and $G(t)$ are matrices describing the contribution of states, controls, and noise to state transition [25.7]. An observation (output) model is also defined in standard state-space form:

$$z(t) = H(t)x(t) + D(t)w(t), \quad (25.11)$$

where $z(t)$ is the observation vector, $w(t)$ is a random variable describing uncertainty in the observation, and where $H(t)$ and $D(t)$ are matrices describing the contribution of state and noise to the observation.

These equations define the evolution of a continuous-time system with continuous observations being made of the state. However, the Kalman Filter is almost always implemented in discrete time $t_k = k$. It is straightforward [25.8] to obtain a discrete-time version of (25.10) and (25.11) in the form

$$x(k) = F(k)x(k-1) + B(k)u(k) + G(k)v(k), \quad (25.12)$$

$$z(k) = H(k)x(k) + D(k)w(k). \quad (25.13)$$

A basic assumption in the derivation of the Kalman filter is that the random sequences $v(k)$ and $w(k)$ describing process and observation noise are all Gaussian,

temporally uncorrelated, and zero-mean

$$E[\mathbf{v}(k)] = E[\mathbf{w}(k)] = \mathbf{0}, \quad \forall k, \quad (25.14)$$

with known covariance

$$E[\mathbf{v}(i)\mathbf{v}^\top(j)] = \delta_{ij}\mathbf{Q}(i), \quad E[\mathbf{w}(i)\mathbf{w}^\top(j)] = \delta_{ij}\mathbf{R}(i). \quad (25.15)$$

It is also generally assumed that the process and observation noises are also uncorrelated

$$E[\mathbf{v}(i)\mathbf{w}^\top(j)] = \mathbf{0}, \quad \forall i, j. \quad (25.16)$$

These are equivalent to a Markov property requiring observations and successive states to be conditionally independent. If the sequences $\mathbf{v}(k)$ and $\mathbf{w}(k)$ are temporally correlated, a shaping filter can be used to whiten the observations, again making the assumptions required for the Kalman filter valid [25.8]. If the process and observation noise sequences are correlated, then this correlation can also be accounted for in the Kalman filter algorithm [25.10]. If the sequence is not Gaussian, but is symmetric with finite moments, then the Kalman filter will still produce good estimates. If however, the sequence has a distribution which is skewed or otherwise pathological, results produced by the Kalman filter will be misleading and there will be a good case for using a more-sophisticated Bayesian filter [25.5].

Filtering Algorithm

The Kalman filter algorithm produces estimates that minimize mean-squared estimation error conditioned on a given observation sequence and so is the conditional mean

$$\hat{\mathbf{x}}(i | j) \triangleq E[\mathbf{x}(i) | \mathbf{z}(1), \dots, \mathbf{z}(j)] \triangleq E[\mathbf{x}(i) | \mathbf{Z}^j]. \quad (25.17)$$

The estimate variance is defined as the mean-squared error in this estimate

$$\mathbf{P}(i | j) \triangleq E\{[\mathbf{x}(i) - \hat{\mathbf{x}}(i | j)][\mathbf{x}(i) - \hat{\mathbf{x}}(i | j)]^\top | \mathbf{Z}^j\}. \quad (25.18)$$

The estimate of the state at a time k given all information up to time k is written $\hat{\mathbf{x}}(k | k)$. The estimate of the state at a time k given only information up to time $k-1$ is called a one-step-ahead prediction (or just a prediction) and is written $\hat{\mathbf{x}}(k | k-1)$.

The Kalman filter algorithm is now stated without proof. Detailed derivations can be found in many books on the subject, for example [25.7,8]. The state is assumed to evolve in time according to (25.12). Observations of this state are made at regular time intervals according

to (25.13). The noise processes entering the system are assumed to obey (25.14), (25.15), and (25.16). It is also assumed that an estimate $\hat{\mathbf{x}}(k-1 | k-1)$ of the state $\mathbf{x}(k-1)$ at time $k-1$ based on all observations made up to and including time $k-1$ is available, and that this estimate is equal to the conditional mean of the true state $\mathbf{x}(k-1)$ conditioned on these observations. The conditional variance $\mathbf{P}(k-1 | k-1)$ in this estimate is also assumed known. The Kalman filter then proceeds recursively in two stages (Fig. 25.3).

Prediction. A prediction $\hat{\mathbf{x}}(k | k-1)$ of the state at time k and its covariance $\mathbf{P}(k | k-1)$ is computed according to

$$\hat{\mathbf{x}}(k | k-1) = \mathbf{F}(k)\hat{\mathbf{x}}(k-1 | k-1) + \mathbf{B}(k)\mathbf{u}(k), \quad (25.19)$$

$$\mathbf{P}(k | k-1) = \mathbf{F}(k)\mathbf{P}(k-1 | k-1)\mathbf{F}^\top(k) + \mathbf{G}(k)\mathbf{Q}(k)\mathbf{G}^\top(k). \quad (25.20)$$

Update. At time k an observation $\mathbf{z}(k)$ is made and the updated estimate $\hat{\mathbf{x}}(k | k)$ of the state $\mathbf{x}(k)$, together with the updated estimate covariance $\mathbf{P}(k | k)$, is computed from the state prediction and observation according to

$$\hat{\mathbf{x}}(k | k) = \hat{\mathbf{x}}(k | k-1) + \mathbf{W}(k)[\mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k | k-1)], \quad (25.21)$$

$$\mathbf{P}(k | k) = \mathbf{P}(k | k-1) - \mathbf{W}(k)\mathbf{S}(k)\mathbf{W}^\top(k), \quad (25.22)$$

where the gain matrix $\mathbf{W}(k)$ is given by

$$\mathbf{W}(k) = \mathbf{P}(k | k-1)\mathbf{H}(k)\mathbf{S}^{-1}(k), \quad (25.23)$$

where

$$\mathbf{S}(k) = \mathbf{R}(k) + \mathbf{H}(k)\mathbf{P}(k | k-1)\mathbf{H}(k) \quad (25.24)$$

is the innovation covariance. The difference between the observation $\mathbf{z}(k)$ and the predicted observation $\mathbf{H}(k)\hat{\mathbf{x}}(k | k-1)$ is termed the *innovation* or residual $\mathbf{v}(k)$:

$$\mathbf{v}(k) = \mathbf{z}(k) - \mathbf{H}(k)\hat{\mathbf{x}}(k | k-1). \quad (25.25)$$

The innovation is an important measure of the deviation between the filter estimates and the observation sequence. Indeed, because the true states are not usually available for comparison with the estimated states, the innovation is often the only measure of how well the estimator is performing. The innovation is particularly important in data association.

The Extended Kalman Filter

The extended Kalman filter (EKF) is a form of the Kalman filter that can be employed when the state model

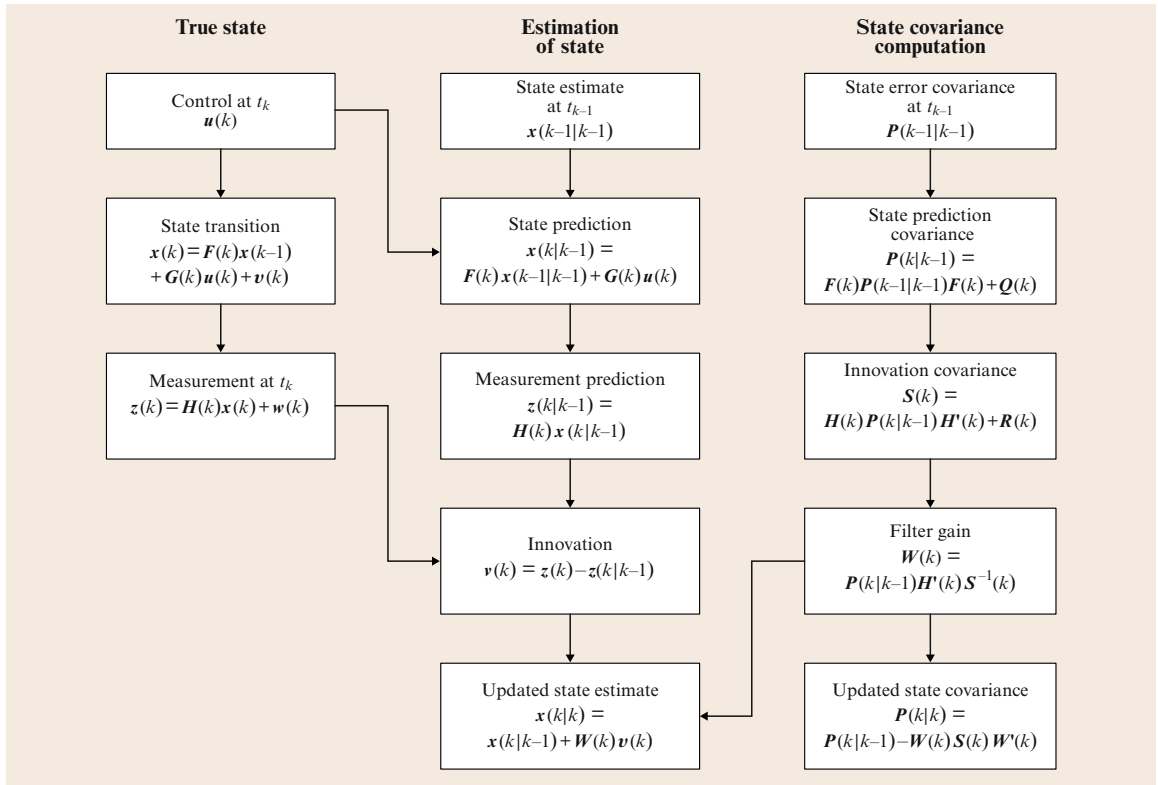


Fig. 25.3 Block diagram of the Kalman filter cycle (after Bar-Shalom and Fortmann [25.7])

and/or the observation model are nonlinear. The **EKF** is briefly described in this section.

The state models considered by the **EKF** are described in state-space notation by a first-order nonlinear vector differential equation or state model of the form

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), \mathbf{v}(t), t], \quad (25.26)$$

where $\mathbf{f}[\cdot, \cdot, \cdot, \cdot]$ is now a general nonlinear mapping of state and control input to state transition. The observation models considered by the **EKF** are described in state-space notation by a nonlinear vector function in the form

$$\mathbf{z}(t) = \mathbf{h}[\mathbf{x}(t), \mathbf{u}(t), \mathbf{w}(t), t], \quad (25.27)$$

where $\mathbf{h}[\cdot, \cdot, \cdot, \cdot]$ is now a general nonlinear mapping of state and control input to observations.

The **EKF**, like the Kalman filter, is almost always implemented in discrete time. By integration and with appropriate identification of discrete time states and observations, the state model is written

$$\mathbf{x}(k) = \mathbf{f}[\mathbf{x}(k-1), \mathbf{u}(k), \mathbf{v}(k), k], \quad (25.28)$$

and the observation model as

$$\mathbf{z}(k) = \mathbf{h}[\mathbf{x}(k), \mathbf{w}(k)]. \quad (25.29)$$

Like the Kalman filter, it is assumed that the noises $\mathbf{v}(k)$ and $\mathbf{w}(k)$ are all Gaussian, temporally uncorrelated, and zero-mean with known variance as defined in (25.14–25.16). The **EKF** aims to minimize the mean-squared error and thereby compute an approximation to the conditional mean. It is assumed therefore that an estimate of the state at time $k-1$ is available which is approximately equal to the conditional mean, $\hat{\mathbf{x}}(k-1 | k-1) \approx \mathbf{E}[\mathbf{x}(k-1) | \mathbf{Z}^{k-1}]$. The **EKF** algorithm will now be stated without proof. Detailed derivations may be found in any number of books on the subject. The principle stages in the derivation of the **EKF** follow directly from those of the linear Kalman filter with the additional step that the process and observation models are linearized as a Taylor series about the estimate and prediction, respectively. The algorithm has two stages: prediction and update.

Prediction. A prediction $\hat{\mathbf{x}}(k | k-1)$ of the state at time k and its covariance $\mathbf{P}(k | k-1)$ is computed according to

$$\hat{\mathbf{x}}(k | k-1) = \mathbf{f}[\hat{\mathbf{x}}(k-1 | k-1), \mathbf{u}(k)], \quad (25.30)$$

$$\mathbf{P}(k | k-1) = \nabla \mathbf{f}_x(k) \mathbf{P}(k-1 | k-1) \nabla^\top \mathbf{f}_x(k) + \nabla \mathbf{f}_v(k) \mathbf{Q}(k) \nabla^\top \mathbf{f}_v(k). \quad (25.31)$$

Update. At time k an observation $\mathbf{z}(k)$ is made and the updated estimate $\hat{\mathbf{x}}(k | k)$ of the state $\mathbf{x}(k)$, together with the updated estimate covariance $\mathbf{P}(k | k)$ is computed from the state prediction and observation according to

$$\hat{\mathbf{x}}(k | k) = \hat{\mathbf{x}}(k | k-1) + \mathbf{W}(k) \{ \mathbf{z}(k) - \mathbf{h}[\hat{\mathbf{x}}(k | k-1)] \}, \quad (25.32)$$

$$\mathbf{P}(k | k) = \mathbf{P}(k | k-1) - \mathbf{W}(k) \mathbf{S}(k) \mathbf{W}^\top(k), \quad (25.33)$$

where

$$\mathbf{W}(k) = \mathbf{P}(k | k-1) \nabla^\top \mathbf{h}_x(k) \mathbf{S}^{-1}(k) \quad (25.34)$$

and

$$\mathbf{S}(k) = \nabla \mathbf{h}_w(k) \mathbf{R}(k) \nabla^\top \mathbf{h}_w(k) + \nabla \mathbf{h}_x(k) \mathbf{P}(k | k-1) \nabla^\top \mathbf{h}_x(k) \quad (25.35)$$

and where the Jacobian $\nabla \mathbf{f}_x(k)$ is evaluated at $\mathbf{x}(k-1) = \hat{\mathbf{x}}(k-1 | k-1)$ and $\nabla \mathbf{h}_x(k)$ is evaluated at $\mathbf{x}(k) = \hat{\mathbf{x}}(k | k-1)$.

A comparison of (25.19–25.24) with (25.30–25.35) makes it clear that the EKF algorithm is very similar to the linear Kalman filter algorithm, with the substitutions $\mathbf{F}(k) \rightarrow \nabla \mathbf{f}_x(k)$ and $\mathbf{H}(k) \rightarrow \nabla \mathbf{h}_x(k)$ being made in the equations for the variance and gain propagation. Thus, the EKF is, in effect, a linear estimator for a state error which is described by a *linear* equation and which is being observed according to a *linear* equation of the form of (25.13).

The EKF works in much the same way as the linear Kalman filter with some notable caveats.

- The Jacobians $\nabla \mathbf{f}_x(k)$ and $\nabla \mathbf{h}_x(k)$ are typically not constant, being functions of both state and time step. This means that unlike the linear filter, the covariances and gain matrix must be computed online as estimates and predictions are made available, and will not in general tend to constant values. This significantly increases the amount of computation which must be performed online by the algorithm.
- As the linearized model is derived by perturbing the true state and observation models around a predicted or nominal trajectory, great care must be taken to ensure that these predictions are always close enough to the true state that second order terms in the linearization are indeed insignificant. If the nominal trajectory is too far away from the true trajectory

then the true covariance will be much larger than the estimated covariance and the filter will become poorly matched. In extreme cases the filter may also become unstable.

- The EKF employs a linearized model that must be computed from an approximate knowledge of the state. Unlike the linear algorithm this means that the filter must be accurately initialized at the start of operation to ensure that the linearized models obtained are valid. If this is not done, the estimates computed by the filter will simply be meaningless.

The Information Filter

The information filter is mathematically equivalent to a Kalman filter. However, rather than generating state estimates $\hat{\mathbf{x}}(i | j)$ and covariances $\mathbf{P}(i | j)$ it uses information state variables $\hat{\mathbf{y}}(i | j)$ and information matrices $\mathbf{Y}(i | j)$, which are related to each other through the relationships

$$\begin{aligned} \hat{\mathbf{y}}(i | j) &= \mathbf{P}^{-1}(i | j) \hat{\mathbf{x}}(i | j), \\ \mathbf{Y}(i | j) &= \mathbf{P}^{-1}(i | j). \end{aligned} \quad (25.36)$$

The information filter has the same prediction–update structure as the Kalman filter.

Prediction. A prediction $\hat{\mathbf{y}}(k | k-1)$ of the information state at time k and its information matrix $\mathbf{Y}(k | k-1)$ is computed according to (Joseph form [25.8]):

$$\begin{aligned} \hat{\mathbf{y}}(k | k-1) &= (\mathbf{I} - \mathbf{\Omega} \mathbf{G}^\top) \mathbf{F}^{-\top} \hat{\mathbf{y}}(k-1 | k-1) \\ &\quad + \mathbf{Y}(k | k-1) \mathbf{B} \mathbf{u}(k), \end{aligned} \quad (25.37)$$

$$\mathbf{Y}(k | k-1) = \mathbf{M}(k) - \mathbf{\Omega} \mathbf{\Sigma} \mathbf{\Omega}^\top, \quad (25.38)$$

respectively, where

$$\begin{aligned} \mathbf{M}(k) &= \mathbf{F}^{-\top} \mathbf{Y}(k-1 | k-1) \mathbf{F}^{-1}, \\ \mathbf{\Sigma} &= \mathbf{G}^\top \mathbf{M}(k) \mathbf{G} + \mathbf{Q}^{-1}, \end{aligned}$$

and

$$\mathbf{\Omega} = \mathbf{M}(t_k) \mathbf{G} \mathbf{\Sigma}^{-1}.$$

It should be noted that $\mathbf{\Sigma}$, whose inverse is required to compute $\mathbf{\Omega}$, has the same dimensionality as the process driving noise, which is normally considerably smaller than the state dimension. Further, the matrix \mathbf{F}^{-1} is the state-transition matrix evaluated backwards in time and so must always exist.

Update. At time k an observation $\mathbf{z}(k)$ is made and the updated information state estimate $\hat{\mathbf{y}}(k | k)$ together with

the updated information matrix $Y(k | k)$ is computed from

$$\hat{y}(k | k) = \hat{y}(k | k-1) + H(k)R^{-1}(k)z(k), \quad (25.39)$$

$$Y(k | k) = Y(k | k-1) + H(k)R^{-1}(k)H^T(k). \quad (25.40)$$

We emphasize that (25.38) and (25.37) are mathematically identical to (25.19) and (25.20), and that (25.39) and (25.40) are mathematically identical to (25.21) and (25.22). It will be noted that there is a duality between information and state space forms [25.10]. This duality is evident from the fact that Ω and Σ in the prediction stage of the information filter play an equivalent role to the gain matrix W and innovation covariance S in the update stage of the Kalman filter. Furthermore, the simple linear update step for the information filter is mirrored in the simple linear prediction step for the Kalman filter.

The main advantage of the information filter over the Kalman filter in data fusion problems is the relative simplicity of the update stage. For a system with n sensors, the fused information state update is *exactly* the linear sum of information contributions from all sensors as

$$\begin{aligned} \hat{y}(k | k) &= \hat{y}(k | k-1) + \sum_{i=1}^n H_i(k)R_i^{-1}(k)z_i(k), \\ Y(k | k) &= Y(k | k-1) + \sum_{i=1}^n H_i(k)R_i^{-1}(k)H_i^T(k). \end{aligned} \quad (25.41)$$

The reason such an expression exists in this form is that the information filter is essentially a log-likelihood expression of Bayes' rule, where products of likelihoods (25.4) are turned into sums. No such simple expression for multisensor updates exists for the Kalman filter. This property of the information filter has been exploited for data fusion in robotic networks [25.11, 12] and more recently in robot navigation and localization problems [25.1]. One substantial disadvantage of the information filter is the coding of nonlinear models, especially for the prediction step.

When to Use a Kalman or Information Filter

Kalman or information filters are appropriate to data fusion problems where the entity of interest is well defined by a continuous parametric state. This would include estimation of the position, attitude, and velocity of a robot or other object, or the tracking of a simple geometric feature such as a point, line or curve. Kalman and information filters are inappropriate for estimating properties

such as spatial occupancy, discrete labels, or processes whose error characteristics are not easily parameterized.

25.1.4 Sequential Monte Carlo Methods

Monte Carlo (MC) filter methods describe probability distributions as a set of weighted samples of an underlying state space. MC filtering then uses these samples to simulate probabilistic inference usually through Bayes' rule. Many samples or simulations are performed. By studying the statistics of these samples as they progress through the inference process, a probabilistic picture of the process being simulated can be built up.

Representing Probability Distributions

In sequential Monte Carlo methods, probability distributions are described in terms of a set of support points (state-space values) x^i , $i = 1, \dots, N$, together with a corresponding set of normalized weights w^i , $i = 1, \dots, N$, where $\sum_i w^i = 1$. The support points and weights can be used to define a probability density function in the form

$$P(x) \approx \sum_{i=1}^N w^i \delta(x - x^i). \quad (25.42)$$

A key question is how these support points and weights are selected to obtain a faithful representation of the probability density $P(x)$. The most general way of selecting support values is to use an *importance density* $q(x)$. The support values x^i are drawn as samples from this density; where the density has high probability, more support values are chosen, and where the density has low probability, few support support vectors are selected. The weights in (25.42) are then computed from

$$w^i \propto \frac{P(x^i)}{q(x^i)}. \quad (25.43)$$

Practically, a sample x^i is drawn from the importance distribution. The sample is then instantiated in the underlying probability distribution to yield the value $P(x = x^i)$. The ratio of the two probability values, appropriately normalized, then becomes the weight.

There are two instructive extremes of the importance sampling method.

1. At one extreme, the importance density could be taken to be a uniform distribution, so the support values x^i are uniformly distributed on the state space in a close approximation to a grid. The probabilities $q(x^i)$ are also therefore equal. The weights computed from (25.43) are then simply proportional to the

probabilities $w^i \propto P(\mathbf{x} = \mathbf{x}^i)$. The result is a model for the distribution which looks very like the regular grid model.

2. At the other extreme, we could choose an importance density equal to the probability model $q(\mathbf{x}) = P(\mathbf{x})$. Samples of the support values \mathbf{x}^i are now drawn from this density. Where the density is high there will be many samples, where the density is low there will be few samples. However, if we substitute $q(\mathbf{x}^i) = P(\mathbf{x}^i)$ into (25.43), it is clear that the weights all become equal $w^i = 1/N$. A set of samples with equal weights is known as a *particle* distribution.

It is, of course, possible to mix these two representations to describe a probability distribution both in terms of a set of weights and in terms of a set of support values. The complete set of samples and weights describing a probability distribution $\{\mathbf{x}^i, w^i\}_{i=1}^N$ is termed a *random measure*.

The Sequential Monte Carlo Method

Sequential Monte Carlo (SMC) filtering is a simulation of the recursive Bayes update equations using sample support values and weights to describe the underlying probability distributions.

The starting point is the recursive or sequential Bayes observation update given in (25.7) and (25.8). The SMC recursion begins with an a posteriori probability density represented by a set of support values and weights $\{\mathbf{x}_{k-1}^i, w_{k-1|k-1}^i\}_{i=1}^{N_{k-1}}$ in the form

$$P(\mathbf{x}_{k-1} | \mathbf{Z}^{k-1}) = \sum_{i=1}^{N_{k-1}} w_{k-1}^i \delta(\mathbf{x}_{k-1} - \mathbf{x}_{k-1}^i). \quad (25.44)$$

The prediction step requires that (25.44) is substituted into (25.8) where the joint density is marginalized. Practically however, this complex step is avoided by implicitly assuming that the importance density is exactly the transition model as

$$q_k(\mathbf{x}_k^i) = P(\mathbf{x}_k^i | \mathbf{x}_{k-1}^i). \quad (25.45)$$

This allows new support values \mathbf{x}_k^i to be drawn on the basis of old support values \mathbf{x}_{k-1}^i while leaving the weights unchanged $w_k^i = w_{k-1}^i$. With this, the prediction becomes

$$P(\mathbf{x}_k | \mathbf{Z}^{k-1}) = \sum_{i=1}^{N_k} w_{k-1}^i \delta(\mathbf{x}_k - \mathbf{x}_k^i). \quad (25.46)$$

The SMC observation update step is relatively straightforward. An observation model $P(z_k | \mathbf{x}_k)$ is defined.

This is a function on both variables, z_k and \mathbf{x}_k , and is a probability distribution on z_k (integrates to unity). When an observation or measurement is made, $z_k = z_k$, the observation model becomes a function of state \mathbf{x}_k only. If samples of the state are taken $\mathbf{x}_k = \mathbf{x}_k^i$, $i = 1 \dots N_k$, the observation model $P(z_k = z_k | \mathbf{x}_k = \mathbf{x}_k^i)$ becomes a set of scalars describing the likelihood that the sample \mathbf{x}_k^i could have given rise to the observation z_k . Substituting these likelihoods and (25.46) into (25.7) gives:

$$\begin{aligned} P(\mathbf{x}_k | \mathbf{Z}^k) \\ = C \sum_{i=1}^{N_k} w_{k-1}^i P(z_k = z_k | \mathbf{x}_k = \mathbf{x}_k^i) \delta(\mathbf{x}_k - \mathbf{x}_k^i). \end{aligned} \quad (25.47)$$

This is normally implemented in the form of an updated set of normalized weights

$$w_k^i = \frac{w_{k-1}^i P(z_k = z_k | \mathbf{x}_k = \mathbf{x}_k^i)}{\sum_{j=1}^{N_k} w_{k-1}^j P(z_k = z_k | \mathbf{x}_k = \mathbf{x}_k^j)} \quad (25.48)$$

and so

$$P(\mathbf{x}_k | \mathbf{Z}^k) = \sum_{i=1}^{N_k} w_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i). \quad (25.49)$$

Note that the support values in (25.49) are the same as those in (25.46), only the weights have been changed by the observation update.

The implementation of the SMC method requires the enumeration of models for both the state transition $P(\mathbf{x}_k | \mathbf{x}_{k-1})$ and the observation $P(z_k | \mathbf{x}_k)$. These need to be presented in a form that allows instantiation of values for z_k , \mathbf{x}_k , and \mathbf{x}_{k-1} . For low-dimensional state spaces, interpolation in a lookup table is a viable representation. For high-dimensional state spaces, the preferred method is to provide a representation in terms of a function.

Practically, (25.46) and (25.49) are implemented as follows.

Time Update. A process model is defined in the usual state-space form as $\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{w}_{k-1}, k)$, where \mathbf{w}_k is an independent noise sequence with known probability density $P(\mathbf{w}_k)$. The prediction step is now implemented as follows: N_k samples \mathbf{w}_k^i , $i = 1, \dots, N_k$ are drawn from the distribution $P(\mathbf{w}_k)$. The N_k support values \mathbf{x}_{k-1}^i together with the samples \mathbf{w}_k^i are passed through the process model as

$$\mathbf{x}_k^i = \mathbf{f}(\mathbf{x}_{k-1}^i, \mathbf{w}_{k-1}^i, k), \quad (25.50)$$

yielding a new set of support vectors \mathbf{x}_k^i . The weights for these support vectors w_{k-1}^i are not changed. In effect, the process model is simply used to do N_k simulations of state propagation.

Observation Update. The observation model is also defined in the usual state-space form as $z_k = \mathbf{h}(\mathbf{x}_k, \mathbf{v}_k, k)$, where \mathbf{v}_k is an independent noise sequence with known probability density $P(\mathbf{v}_k)$. The observation step is now implemented as follows. A measurement $z_k = z_k$ is made. For each support value \mathbf{x}_k^i , a likelihood is computed as

$$\Lambda(\mathbf{x}_k^i) = P(z_k = z_k | \mathbf{x}_k = \mathbf{x}_k^i). \quad (25.51)$$

Practically, this requires that the observation model be in an equational form (such as a Gaussian) which allows computation of the likelihood in the error between the measured value z_k and the observations predicted by each particle $\mathbf{h}(\mathbf{x}_k^i, k)$. The updated weights after observation are just

$$w_k^i \propto w_{k-1}^i P(z_k = z_k | \mathbf{x}_k^i). \quad (25.52)$$

Resampling

After the weights are updated it is usual to resample the measure $\{\mathbf{x}_k^i, w_k^i\}_{i=1}^N$. This focuses the samples in on those areas that have most probability density. The decision to resample is made on the basis of the *effective number*, N_{eff} of particles in the sample, approximately estimated from $N_{\text{eff}} = 1 / \sum_i (w_k^i)^2$. The sampling importance resampling (SIR) algorithm resamples at every cycle so that the weights are always equal. One of the key problems with resampling is that the sample set fixates on a few highly likely samples. This problem of fixating on a few highly likely particles during resampling is known as *sample impoverishment*. Generally, it is good to resample when N_{eff} falls to some fraction of the actual samples (say 1/2).

When to Use Monte Carlo Methods

Monte Carlo (MC) methods are well suited to problems where state-transition models and observation models are highly nonlinear. This is because sample-based methods can represent very general probability densities. In particular multimodal or multiple hypothesis density functions are well handled by Monte Carlo techniques. One caveat to note however is that the models $P(\mathbf{x}_k | \mathbf{x}_{k-1})$ and $P(z_k | \mathbf{x}_k)$ must be enumerable in all cases and must typically be in a simple parametric form. MC methods also span the gap between parametric and grid-based data fusion methods.

Monte Carlo methods are inappropriate for problems where the state space is of high dimension. In general the number of samples required to faithfully model a given density increases exponentially with state-space dimension. The effects of dimensionality can be limited by marginalizing out states that can be modeled without sampling, a procedure known as Rao–Blackwellization.

25.1.5 Alternatives to Probability

The representation of uncertainty is so important to the problem of information fusion that a number of alternative modeling techniques have been proposed to deal with perceived limitations in probabilistic methods.

There are three main perceived limitations of probabilistic modeling techniques.

1. Complexity: the need to specify a large number of probabilities to be able to apply probabilistic reasoning methods correctly.
2. Inconsistency: the difficulties involved in specifying a consistent set of beliefs in terms of probability and using these to obtain consistent deductions about states of interest.
3. Precision of models: the need to be precise in the specification of probabilities for quantities about which little is known.
4. Uncertainty about uncertainty: the difficulty in assigning probability in the face of uncertainty, or ignorance about the source of information.

There are three main techniques put forward to address these issues: interval calculus, fuzzy logic, and the theory of evidence (Dempster–Shafer methods). We briefly discuss each of these in turn.

Interval Calculus

The representation of uncertainty using an interval to bound true parameter values has a number of potential advantages over probabilistic techniques. In particular, intervals provide a good measure of uncertainty in situations where there is a lack of probabilistic information, but in which sensor and parameter error is known to be bounded. In interval techniques, the uncertainty in a parameter x is simply described by a statement that the true value of the state x is known to be bounded from below by a , and from above by b ; $x \in [a, b]$. It is important that no other additional probabilistic structure is implied; in particular the statement $x \in [a, b]$ does not necessarily imply that x is equally probable (uniformly distributed) over the interval $[a, b]$.

There are a number of simple and basic rules for the manipulation of interval errors. These are described in detail in the book by Moore [25.13] (whose analysis was originally aimed at understanding limited-precision computer arithmetic). Briefly, with $a, b, c, d \in \mathbb{R}$, addition, subtraction, multiplication, and division are defined by the following algebraic relations

$$\begin{aligned} [a, b] + [c, d] &= [a + c, b + d], \\ [a, b] - [c, d] &= [a - d, b - c], \end{aligned} \quad (25.53)$$

$$[a, b] \times [c, d] = [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)], \quad (25.54)$$

$$\frac{[a, b]}{[c, d]} = [a, b] \times \left[\frac{1}{d}, \frac{1}{c} \right], \quad 0 \notin [c, d]. \quad (25.55)$$

It can be shown that interval addition and multiplication are both associative and commutative. Interval arithmetic admits an obvious metric distance measure:

$$d([a, b], [c, d]) = \max(|a - c|, |b - d|). \quad (25.56)$$

Matrix arithmetic using intervals is also possible, but substantially more complex, particularly when matrix inversion is required.

Interval calculus methods are sometimes used for detection. However, they are not generally used in data fusion problems since:

1. it is difficult to get results that converge to anything of value (it is too pessimistic), and
2. it is hard to encode dependencies between variables which are at the core of many data fusion problems.

Fuzzy Logic

Fuzzy logic has found widespread popularity as a method for representing uncertainty particularly in applications such as supervisory control and high-level data fusion tasks. It is often claimed that fuzzy logic provides an ideal tool for inexact reasoning, particularly in rule-based systems. Certainly, fuzzy logic has had some notable success in practical application.

A great deal has been written about fuzzy sets and fuzzy logic (see, for example, [25.14] and the discussion in [25.15, Chap. 11]). Here we briefly describe the main definitions and operations without any attempt to consider the more advanced features of fuzzy-logic methods.

Consider a universal set consisting of the elements x ; $\mathcal{X} = \{x\}$. Consider a proper subset $\mathcal{A} \subseteq \mathcal{X}$ such that

$$\mathcal{A} = \{x \mid x \text{ has some specific property}\}.$$

In conventional logic systems, we can define a membership function $\mu_A(x)$ (also called the *characteristic function* which reports if a specific element $x \in \mathcal{X}$ is a member of this set:

$$\mathcal{A} \Rightarrow \mu_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if } x \notin A \end{cases}.$$

For example \mathcal{X} may be the set of all aircraft. The set \mathcal{A} may be the set of all supersonic aircraft. In the fuzzy logic literature, this is known as a *crisp* set. In contrast, a fuzzy set is one in which there is a *degree of membership*, ranging between 0 and 1. A fuzzy membership function $\mu_A(x)$ then defines the degree of membership of an element $x \in \mathcal{X}$ to the set \mathcal{A} . For example, if \mathcal{X} is again the set of all aircraft, \mathcal{A} may be the set of all *fast* aircraft. Then the fuzzy membership function $\mu_A(x)$ assigns a value between 0 and 1 indicating the degree of membership of every aircraft x to this set. Formally

$$\mathcal{A} \Rightarrow \mu_A \mapsto [0, 1].$$

Composition rules for fuzzy sets follow the composition processes for normal crisp sets, for example,

$$\mathcal{A} \cap \mathcal{B} \Rightarrow \mu_{A \cap B}(x) = \min[\mu_A(x), \mu_B(x)],$$

$$\mathcal{A} \cup \mathcal{B} \Rightarrow \mu_{A \cup B}(x) = \max[\mu_A(x), \mu_B(x)].$$

The normal properties associated with binary logic now hold: commutativity, associativity, idempotence, distributivity, De Morgan's law, and absorption. The only exception is that the law of the excluded middle is no longer true

$$\mathcal{A} \cup \overline{\mathcal{A}} \neq \mathcal{X}, \quad \mathcal{A} \cap \overline{\mathcal{A}} \neq \emptyset.$$

Together these definitions and laws provide a systematic means of reasoning about inexact values.

The relationship between fuzzy set theory and probability is still hotly debated.

Evidential Reasoning

Evidential reasoning (often called the Dempster–Shafer theory of evidence after the originators of these ideas) has seen intermittent success particularly in automated reasoning applications. Evidential reasoning is qualitatively different from either probabilistic methods or fuzzy set theory in the following sense. Consider a universal set \mathcal{X} . In probability theory or fuzzy set theory, a belief mass may be placed on any element $x_i \in \mathcal{X}$

and indeed on any subset $\mathcal{A} \subseteq \mathcal{X}$. In evidential reasoning, belief mass can not only be placed on elements and sets, but also sets of sets. Specifically, while the domain of probabilistic methods is all possible subsets \mathcal{X} , the domain of evidential reasoning is the power set $2^{\mathcal{X}}$.

As an example, consider the mutually exclusive set $\mathcal{X} = \{\text{occupied}, \text{empty}\}$. In probability theory we might assign a probability to each possible event, for example, $P(\text{occupied}) = 0.3$, and thus $P(\text{empty}) = 0.7$. In evidential reasoning, we construct the set of all subsets

$$2^{\mathcal{X}} = \{\{\text{occupied}, \text{empty}\}, \{\text{occupied}\}, \{\text{empty}\}, \emptyset\},$$

and belief mass is assigned to all elements of this set as

$$\begin{aligned} m(\{\text{occupied}, \text{empty}\}) &= 0.5, \\ m(\{\text{occupied}\}) &= 0.3, \\ m(\{\text{empty}\}) &= 0.2, \\ m(\emptyset) &= 0.0, \end{aligned}$$

(the null set \emptyset is assigned a belief mass of zero for normalization purposes). The interpretation of this is that there is a 30% chance of occupied, a 20% chance of empty, and a 50% chance of either occupied or empty. In effect, the measure placed on the set containing both occupied and empty is a measure of ignorance or inability to distinguish between the two alternatives. See [25.16] for a more-detailed example of applying the evidential method to certainty-grid navigation.

Evidential reasoning thus provides a method of capturing ignorance or an inability to distinguish between alternatives. In probability theory, this would be dealt with in a very different manner by assigning an equal or uniform probability to each alternative. Yet, stating that there is a 50% chance of occupancy is clearly *not* the same as saying that it is unknown whether it will be occupied or not. The use of the power set as the *frame of discernment* allows a far richer representation of beliefs. However, this comes at the cost of a substantial increase in complexity. If there are n elements in the original set \mathcal{X} , then there will be 2^n possible subsets on which a belief mass will be assigned. For large n , this is clearly intractable. Further, when the set is continuous, the set of all subsets is not even measurable.

Evidential reasoning methods provide a means of assigning, and combining belief masses on sets. Methods also exist for obtaining related measures called *support* and *plausibility* which, in effect, provide upper and lower probability bounds in agreement with Dempster's original formulation of this method.

Evidential reasoning can play an important role in discrete data fusion systems, particularly in areas such as attribute fusion and situation assessment, where information may be unknown or ambiguous. Its use in lower-level data fusion problems is challenging as the assignment of belief mass to the power set scales exponentially with state cardinality.

25.2 Multisensor Fusion Architectures

The multisensor fusion methods described in the previous section provide the algorithmic means by which sensor data and their associated uncertainty models can be used to construct either implicit or explicit models of the environment. However, a multisensor fusion system must include many other functional components to manage and control the fusion process. The organization of these is termed a *multisensor fusion architecture*.

25.2.1 Architectural Taxonomy

Multisensor systems architectures can be organized in various ways. The military community has developed a layout of functional architectures based on the joint directors of the laboratories (JDL) model for multisensor systems. This approach views multisensor fusion in terms of signal, feature, threat and situation analysis levels (so-called JDL levels). The assessment of such

systems is specified in terms of tracking performance, survivability, efficiency and bandwidth. Such measures are not generally appropriate in robotics applications and so the JDL model is not discussed further here (see [25.17, 18] for details). Other classification schemes distinguish between low- and high-level fusion [25.19], or centralized versus decentralized processing or data versus variable [25.20].

A general architectural framework for multisensor robotic systems has been developed and described in detail by Makarenko [25.21], and we will base our discussion on this approach. A system architecture is defined as follows.

Meta-architecture. A set of high-level considerations that strongly characterize the system structure. The selection and organization of the system elements may be guided by aesthetics, efficiency, or other design cri-

teria and goals (for example, system and component comprehensibility, modularity, scalability, portability, interoperability, (de)centralization, robustness, fault tolerance).

Algorithmic Architecture. A specific set of information fusion and decision-making methods. These methods address data heterogeneity, registration, calibration, consistency, information content, independence, time interval and scale, and relationships between models and uncertainty.

Conceptual Architecture. The granularity and functional roles of components (specifically, mappings from algorithmic elements to functional structures).

Logical Architecture. Detailed canonical component types (i. e., object-oriented specifications) and interfaces to formalize intercomponent services. Components may be ad hoc or regimented, and other concerns include granularity, modularity, reuse, verification, data structures, semantics, etc. Communication issues include hierarchical versus heterarchical organization, shared memory versus message passing, information-based characterizations of subcomponent interactions, pull/push mechanisms, subscribe–publish mechanisms, etc. Control involves both the control of actuation systems within the multisensor fusion system, as well as control of information requests and dissemination within the system, and any external control decisions and commands.

Execution Architecture. Defines mapping of components to execution elements. This includes internal or external methods of ensuring correctness of the code (i. e., that the environment and sensor models have been correctly transformed from mathematical or other formal descriptions into computer implementations), and also validation of the models (i. e., ensure that the formal descriptions match physical reality to the required extent).

In any closed-loop control system, sensors are used to provide the feedback information describing the current status of the system and its uncertainties. Building a sensor system for a given application is a system engineering process that includes the analysis of system requirements, a model of the environment, the determination of system behavior under different conditions, and the selection of suitable sensors [25.22]. The next step in building the sensor system is to assemble the hardware components and develop the necessary

software modules for data fusion and interpretation. Finally, the system is tested, and the performance is analyzed. Once the system is built, it is necessary to monitor the different components of the system for the purpose of testing, debugging, and analysis. The system also requires quantitative measures in terms of time complexity, space complexity, robustness, and efficiency.

In addition, designing and implementing real-time systems are becoming increasingly complex owing to many added features such as graphical user interfaces (GUIs), visualization capabilities, and the use of many sensors of different types. Therefore, many software engineering issues such as reusability and the use of commercial off-the-shelf (COTS) components [25.23], real-time issues [25.24–26], sensor selection [25.27], reliability [25.28–30], and embedded testing [25.31] are now receiving more attention from system developers.

Each sensor type has different characteristics and functional descriptions. Consequently, some approaches aim to develop general methods of modeling sensor systems in a manner that is independent of the physical sensors used. In turn, this enables the performance and robustness of multisensor systems to be studied in a general way. There have been many attempts to provide *the* general model, along with its mathematical basis and description. Some of these modeling techniques concern error analysis and fault tolerance of multisensor systems [25.32–37]. Other techniques are model based, and require a priori knowledge of the sensed object and its environment [25.38–40]. These help fit data to a model, but do not always provide the means to compare alternatives. Task-directed sensing is another approach to devising sensing strategies [25.41–43]. General sensor modeling work has had a considerable influence on the evolution of multisensor fusion architectures.

Another approach to modeling sensor systems is to define sensori-computational systems associated with each sensor to allow design, comparison, transformation, and reduction of any sensory system [25.44]. In this approach, the concept of an information invariant is used to define a measure of information complexity. This provides a computational theory allowing analysis, comparison, and reduction of sensor systems.

In general terms, multisensor fusion architectures may be classified according to the choice along four independent design dimensions:

1. centralized–decentralized,
2. local–global interaction of components,

3. modular–monolithic, and
4. heterarchical–hierarchical.

The most prevalent combinations are:

- centralized, global interaction, and hierarchical,
- decentralized, global interaction, and heterarchical,
- decentralized, local interaction, and hierarchical,
- decentralized, local interaction, and heterarchical.

In some cases explicit modularity is also desirable. Most existing multisensor architectures fit reasonably well into one of these categories. These categories make no general commitment to the algorithmic architecture. If the algorithmic architecture is the predominant feature of a system, it will be characterized as part of multisensor fusion theory in Sect. 25.1; otherwise, it merely differentiates methods within one of the four meta-architectures.

25.2.2 Centralized, Local Interaction, and Hierarchical

Centralized, local interaction and hierarchical architectures encompass a number of system philosophies. Least representationally demanding is the *subsumption architecture* initially proposed by *Braitenberg* [25.45] and popularized by *Brooks* [25.46]. The subsumption multisensor architecture defines behaviors as the basic components, and employs a layered set of behaviors to embody one program (monolithic). Any behavior may utilize the output of other behaviors, and may also inhibit other behaviors. The hierarchy is defined by the layers, although this is not always clear-cut. The major design philosophy is to develop behaviors directly from perception–action loops without recourse to brittle, environment representations. This leads to robustness in operation, but a lack of composite behavior predictability.

A more sophisticated (representationally) behavior-based system is the *distributed field robot architecture* (DFRA) [25.47]. This is a generalization of the sensor fusion effects (SFX) architecture [25.48]. This approach exploits modularity, and aims to achieve both behavior-based and deliberative action, reconfigurability and interoperability through the use of Java, Jini, and XML, fault tolerance, adaptability, longevity, consistent interfaces, and dynamic components. The algorithmic architecture is based on fuzzy logic controllers. Experiments have been demonstrated on outdoor mobile robot navigation.

Other similar architectures of this type include perception action networks *Lee* [25.49, 50], while *Draper* [25.51] focuses on types of information needed to perform tasks (higher-level integration); see also [25.52].

Another approach to this type of sensor fusion is to use artificial neural networks. The advantage is that the user, at least in principle, does not need to understand how sensor modalities relate, nor model the uncertainties, nor in fact determine the structure of the system more than to specify the number of layers in the network and the number of nodes per layer. The neural network is presented with a set of training examples, and must determine through the weights on the neuron connections the optimal mapping from inputs to desired outputs (classifications, control signals, etc.) [25.53, 54].

Various other methods exist; for example, *Hager* [25.42, 43] defines a task-oriented approach to sensor fusion based on Bayesian decision theory and develops an object-oriented programming framework. *Joshi* and *Sanderson* [25.55] describe a

“methodology for addressing model selection and multisensor fusion issues using representation size (description length) to choose (1) model class and number of parameters, (2) model parameter resolution (3) subset of observed features to model, and (4) correspondence to map features to models.”

Their approach is broader than an architecture and uses a minimization criterion to synthesize a multisensor fusion system to solve specific two-dimensional (2-D) and three-dimensional (3-D) object recognition problems.

25.2.3 Decentralized, Global Interaction, and Heterarchical

The major example of the decentralized, global interaction meta-architecture is the blackboard system. There have been many examples of blackboard systems developed for data fusion applications. For example, the SEPIA system of *Cherfaoui* and *Vachon* [25.56] uses logical sensors (see below) in the form of modular agents that post results to a blackboard. The overall architectural goals for blackboards include efficient collaboration and dynamic configuration. Experiments are reported on an indoor robot moving from room to room.

The MESSIE (Multi Expert System for Scene Interpretation and Evaluation) system [25.57] is a scene interpretation system based on multisensor fusion; it has been applied to the interpretation of remotely sensed im-

ages. A typology of the multisensor fusion concepts is presented, and the consequences of modeling problems for objects, scene, and strategy are derived. The proposed multispecialist architecture generalized the ideas of their previous work by taking into account the knowledge of sensors, the multiple viewing notion (shot), and the uncertainty and imprecision of models and data modeled with possibility theory. In particular, generic models of objects are represented by concepts independent of sensors (geometry, materials, and spatial context). Three kinds of specialists are present in the architecture: generic specialists (scene and conflict), semantic object specialists, and low-level specialists. A black-board structure with a centralized control is used. The interpreted scene is implemented as a matrix of pointers enabling conflicts to be detected very easily. Under the control of the scene specialist, the conflict specialist resolves conflicts using the spatial context knowledge of objects. Finally, an interpretation system with SAR (synthetic aperture radar)/SPOT sensors is described, and an example of a session concerned with bridge, urban area, and road detection is shown.

25.2.4 Decentralized, Local Interaction, and Hierarchical

One of the earliest proposals for this type of architecture is the real-time control system (RCS) [25.58]. RCS is presented as a cognitive architecture for intelligent control, but essentially uses multisensor fusion to achieve complex control. RCS focuses on task decomposition as the fundamental organizing principle. It defines a set of nodes, each comprised of a sensor processor, a world model, and a behavior generation component. Nodes communicate with other nodes, generally in a hierarchical manner, although across-layer connections are allowed. The system supports a wide variety of algorithmic architectures, from reactive behavior to semantic networks. Moreover, it maintains signals, images, and maps, and allows tight coupling between iconic and symbolic representations. The architecture does not generally allow dynamic reconfiguration, but maintains the static module connectivity structure of the specification. RCS has been demonstrated in unmanned ground vehicles [25.59]. Other object-oriented approaches have been reported [25.34, 60].

An early architectural approach which advocated strong programming semantics for multisensor systems is the logical sensor system (LSS). This approach exploits functional (or applicative) language theory to achieve that.

The most developed version of LSS is instrumented LSS (ILSS) [25.22]. The ILSS approach is based on the LSS introduced by *Shilcrat* and *Henderson* [25.61]. The LSS methodology is designed to specify any sensor in a way that hides its physical nature. The main goal behind LSS was to develop a coherent and efficient presentation of the information provided by many sensors of different types. This representation provides a means for recovery from sensor failure, and also facilitates reconfiguration of the sensor system when adding or replacing sensors [25.62].

ILSS is defined as an extension to LSS, and is comprised of the following components (Fig. 25.4):

1. *ILS name*: uniquely identifies a module
2. *Characteristic output vector (COV)*: strongly typed output structure, with one output vector and zero or more input vectors
3. *Commands*: input commands to the module, and output commands to the other modules
4. *Select function*: a selector that detects the failure of an alternate and switches to another alternate if possible
5. *Alternate subnets*: alternative ways of producing the COV_{out} ; it is these implementations of one or more algorithms that carry the main functions of the module
6. *Control command interpreter (CCI)*: interpreter of the commands to the module
7. *Embedded tests*: self-testing routines that increase robustness and facilitate debugging

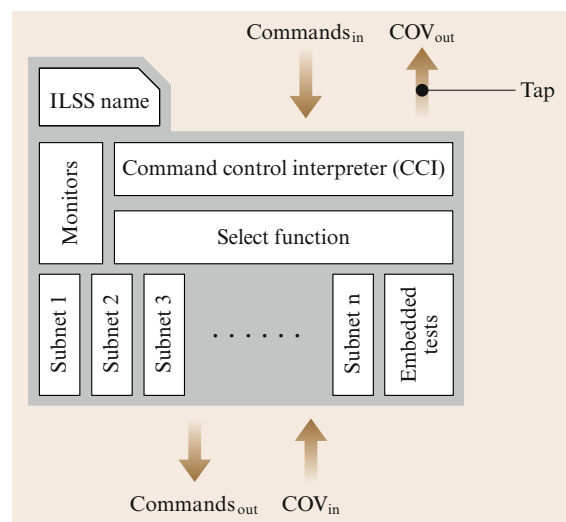


Fig. 25.4 Instrumented logical sensor module

8. *Monitors*: modules that check the validity of the resulting COVs
9. *Taps*: hooks on the output lines to view different COV values

These components identify the system behavior and provide mechanisms for online monitoring and debugging. In addition, they provide handles for measuring the runtime performance of the system. Monitors are validity check stations that filter the output and alert the user to any undesired results. Each monitor is equipped with a set of rules (or constraints) that governs the behavior of the COV under different conditions.

Embedded testing is used for online checking and debugging purposes. *Weller* proposed a sensor-processing model with the ability to detect measurement errors and to recover from these errors [25.31]. This method is based on providing each system module with verification tests to verify certain characteristics in the measured data, and to verify the internal and output data resulting from the sensor-module algorithm. The recovery strategy is based on rules that are local to the different sensor modules. ILSS uses a similar approach called *local embedded testing*, in which each module is equipped with a set of tests based on the semantic definition of that module. These tests generate input data to check different aspects of the module, then examine the output of the module using a set of constraints and rules defined by the semantics. These tests can also take input from other modules to check the operation of a group of modules. Examples are given of a wall-posed estimation system comprised of a Labmate platform with a camera and sonars. Many extensions have been proposed for LSS [25.63, 64].

25.2.5 Decentralized, Local Interaction, and Heterarchical

The best example of this meta-architecture is the active sensor network (ASN) framework for distributed data fusion developed by *Makarenko* [25.21, 65]. The distinguishing features of the various architectures are now described.

Meta-architecture

The distinguishing features of ASN are its commitment to decentralization, modularity, and strictly local interactions (this may be physical or by type). Thus, these are communicating processes. Decentralized means that no component is central to operation of the system, and the communication is peer to peer. Also, there are no central facilities or services (e.g., for communication, name and service lookup or timing). These features lead to a system that is scalable, fault tolerant, and reconfigurable.

Local interactions mean that the number of communication links does not change with the network size. Moreover, the number of messages should also remain constant. This makes the system scalable as well as reconfigurable.

Modularity leads to interoperability derived from interface protocols, reconfigurability, and fault tolerance: failure may be confined to individual modules.

Algorithmic Architecture

There are three main algorithmic components: belief fusion, utility fusion, and policy selection. Belief fusion is achieved by communicating all beliefs to neighboring platforms. A belief is defined as a probability distribution of the world state space.

Table 25.1 Canonical components and the roles they play. Multiple X’s in the same row indicate that some interrole relationships are internalized within a component. ‘Frame’ does not participate in information fusion or decision making but is required for localization and other platform-specific tasks (from [25.21])

Component Type	Belief			Plan			Action	
	Source	Fuse/Dist	Sink	Source	Fuse/Dist	Sink	Source	Sink
Sensor	x							
Node		x			x			
Actuator								x
Planner			x	x		x	x	
UI	x		x			x	x	
Frame								

Utility fusion is handled by separating the individual platform's partial utility into the team utility of belief quality and local utilities of action and communication. The downside is that the potential coupling between individual actions and messages is ignored because the utilities of action and communication remain local.

The communication and action policies are chosen by maximizing expected values. The selected approach is to achieve point maximization for one particular state and follows the work of *Manyika* and *Grocholsky* [25.11, 66].

Conceptual Architecture

The data types of the system include

1. *beliefs*: current world beliefs
2. *plans*: future planned world beliefs
3. *actions*: future planned actions

The definition of component roles leads to a natural partition of the system.

The information fusion task is achieved through the definition of four component roles for each data type: *source*, *sink*, *fuser*, and *distributor*. (Note that the data type action does not have fuser or distributor component roles.)

Connections between distributors form the backbone of the *ASN* framework, and the information exchanged is in the form of their local beliefs. Similar consider-

ations are used to determine component roles for the decision-making and system configuration tasks.

Logical Architecture

A detailed architecture specification is determined from the conceptual architecture. It is comprised of six canonical component types as described in Table 25.1 [25.21].

Makarenko then describes how to combine the components and interfaces to realize the use cases of the problem domain in *ASN*.

Execution Architecture

The execution architecture traces the mapping of logical components to runtime elements such as processes and shared libraries. The deployment view shows the mapping of physical components onto the nodes of the physical system. The source code view explains how the software implementing the system is organized. At the architectural level, three items are addressed: execution, deployment, and source code organization.

The experimental implementation of the *ASN* framework has proven to be flexible enough to accommodate a variety of system topologies, platform, and sensor hardware, and environment representations. Several examples are given with a variety of sensors, processors, and hardware platforms.

25.3 Applications

Multisensor fusion systems have been applied to a wide variety of problems in robotics (see the references for this chapter), but the two most general areas are *dynamic system control* and *environment modeling*. Although there is some overlap between these, they may generally be characterized as

- *Dynamic system control*: the problem is to use appropriate models and sensors to control the state of a dynamic system (e.g., industrial robot, mobile robot, autonomous vehicle, surgical robot). Usually such systems involve real-time feedback control loops for steering, acceleration, and behavior selection. In addition to state estimation, uncertainty models are required. Sensors may include force/torque sensors, gyros, global positioning system (*GPS*), position encoders, cameras, range finders, etc.;
- *environment modeling*: the problem is to use appropriate sensors to construct a model of some aspect of the physical environment. This may be a particular object, e.g., a cup, a physical part, a face, or a larger part of the surroundings, e.g., the interior of a building, part of a city or an extended remote or underground area. Typical sensors include cameras, radar, 3-D range finders, infrared (*IR*), tactile sensors and touch probes (*CMMs*), etc. The result is usually expressed as geometry (points, lines, surfaces), features (holes, sinks, corners, etc.), or physical properties. Part of the problem includes the determination of optimal sensor placement.

25.3.1 Dynamic System Control

The *EMS-Vision* system [25.67] is an outstanding exemplar of this application domain. The goal is to develop



Fig. 25.5 First fully autonomous vehicle on German autobahn

a robust and reliable perceptual system for autonomous vehicles. The development goals, as stated by the EMS-Vision team, are:

- COTS components
- wide variety of objects modeled and incorporated into behaviors
- inertial sensors for ego-state estimation
- peripheral/foveal/saccadic vision

- knowledge and goal driven behavior
- state tracking for objects
- 25 Hz real-time update rate

The approach has been in development since the 1980s. Figure 25.5 shows the first vehicle to drive fully autonomously on the German autobahn for 20 km and at speeds up to 96 km/h.

Information from inertial and vision sensors is combined to produce a road scene tree (Fig. 25.6). A four-dimensional (4-D) generic object representation is built which includes background knowledge of the object (e.g., roads), its behavioral capabilities, object state and variances, and shape and aspect parameters. Figure 25.7 shows the 4-D inertial/vision multisensor guidance system, while Fig. 25.8 shows the hardware aspects.

In summary, the EMS-Vision system is an interesting and powerful demonstration of multisensor fusion for dynamic system control.

25.3.2 ANSER II: Decentralized Data Fusion

Decentralized data fusion (DDF) methods were initially motivated by the insight that the information or canonical form of the conventional Kalman filter data fusion algorithm could be implemented by simply adding information contributions from observations as shown in (25.41). As these (vector and

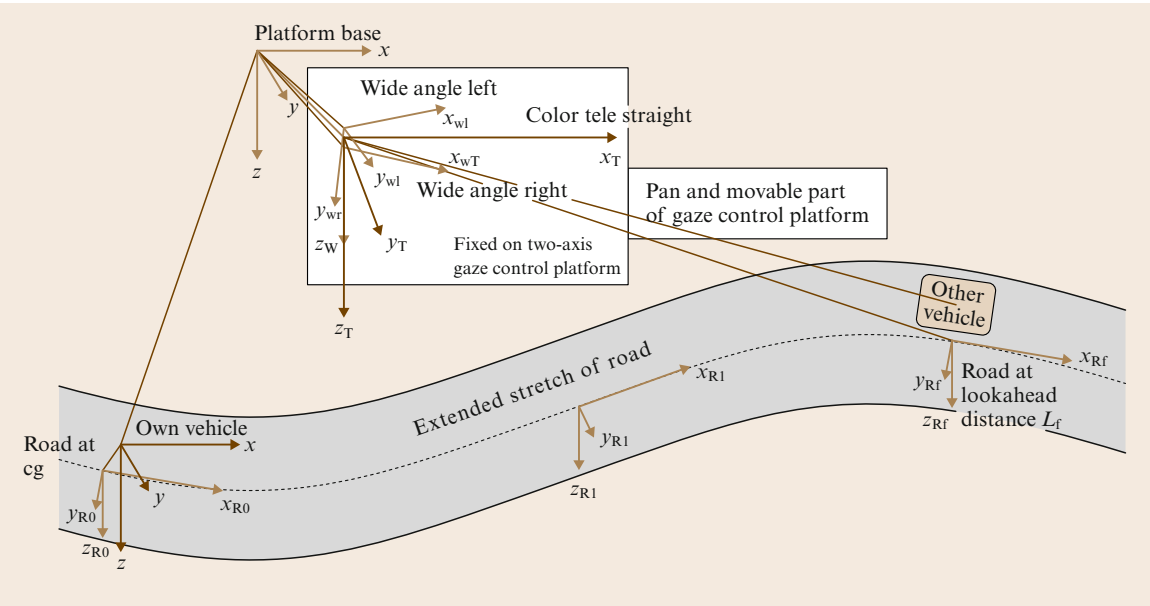


Fig. 25.6 EMS-Vision road scene tree

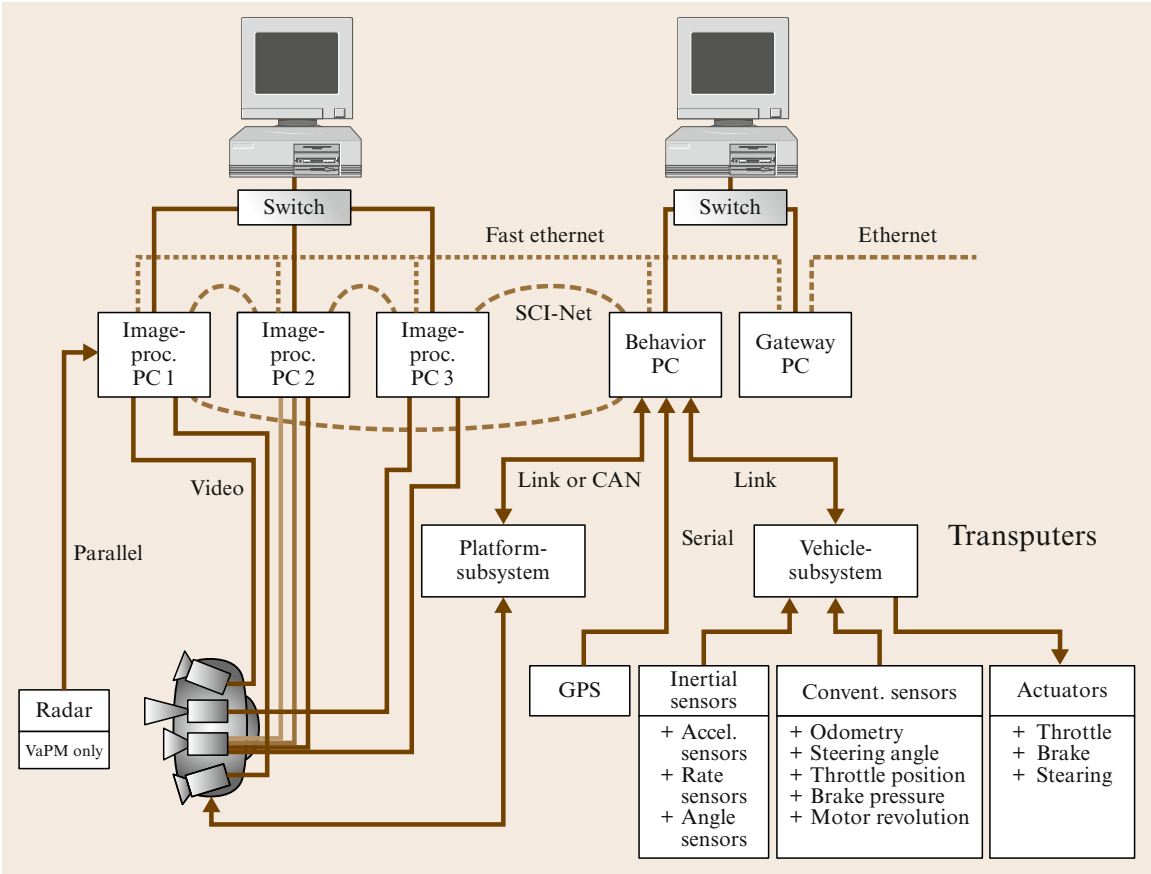


Fig. 25.8 EMS-Vision hardware layout (CAN: controller area network)

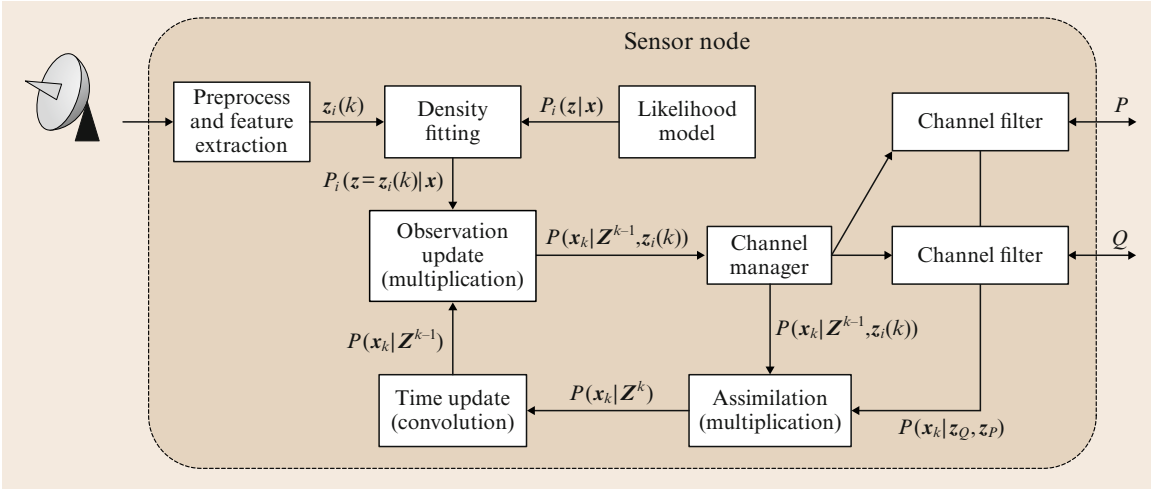


Fig. 25.9 Mathematical structure of a decentralized data fusion node

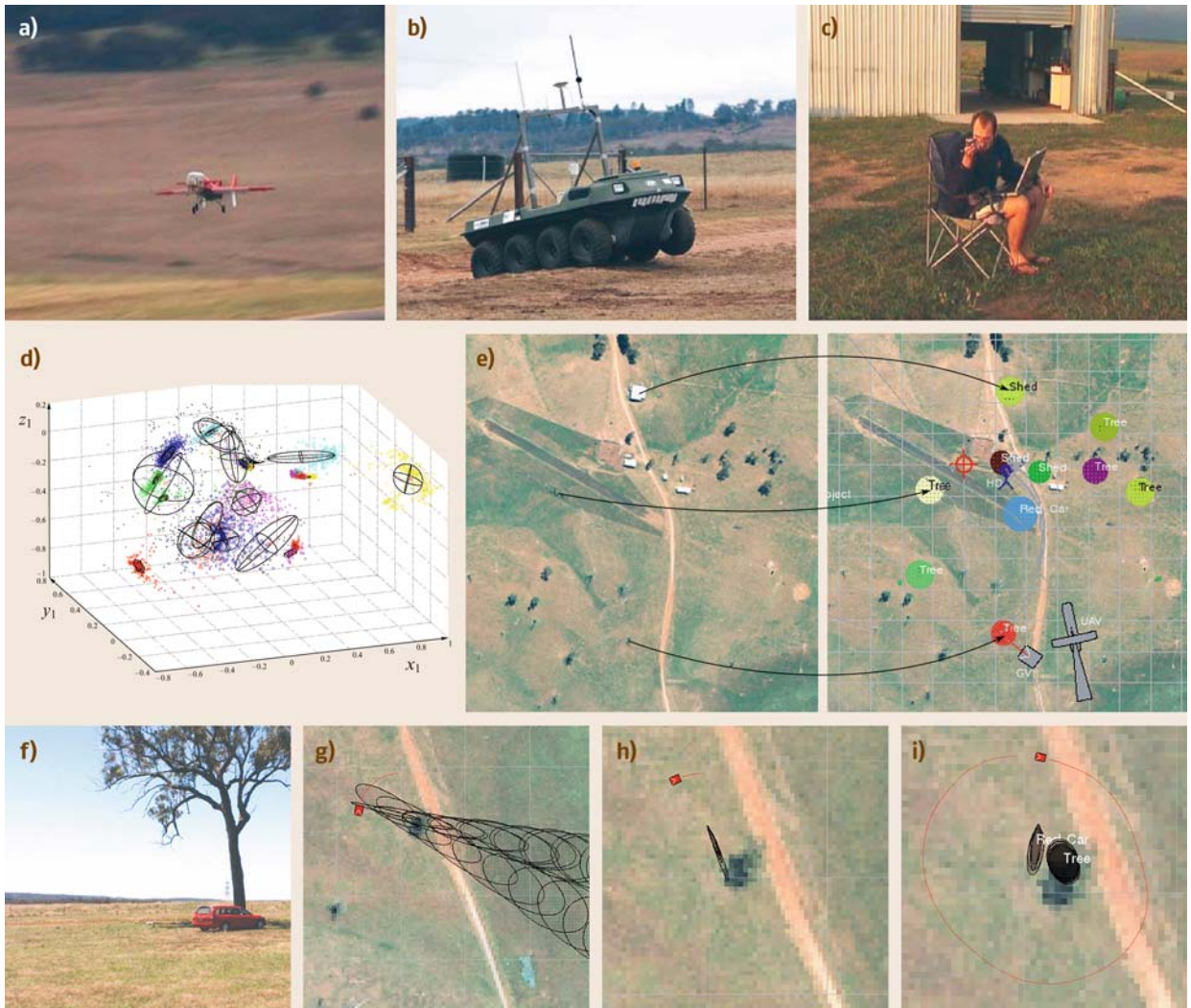


Fig. 25.10a–i A synopsis of the ANSER II autonomous network and its operation. (a–c) Main system components: (a) air vehicle, (b) ground vehicle, (c) human operative. (d–e) The perception process: (d) top three dimensions of features discovered from ground-based visual sensor data along with the derived mixture model describing these feature properties, (e) sector of the overall picture obtained from fusing air vehicle (UAV), ground vehicle (GV), and human operator (HO) information. Each set of ellipses corresponds to a particular feature and the labels represent the identity state with highest probability. (f–i) Sequential fusion process for two close landmarks: (f) a tree and a red car, (g) bearing-only visual observations of these landmarks are successively fused, (h) to determine location and identity (i). Note the Gaussian mixture model for the bearing measurement likelihood

network. Multiple observations of the same feature, possibly by different platforms, result in an increasingly accurate estimate of the feature location for all nodes. A corresponding discrete probability measure is used for Fig. 25.10 shows a synopsis of the operation of the ANSER II system.

The ANSER II system demonstrates a number of general principles in Bayesian data fusion methods, specifically the need to model sensors appropriately through the likelihood function, and the possibility of building very different data fusion architectures from the essential Bayesian form.

25.4 Conclusions

Multisensor data fusion has progressed greatly in the last few decades; further advances in the field will be documented in the robotics and multisensor fusion and integration conference and journal literature. Robust applications are being fielded based on the body of theory and experimental knowledge produced by the research community. Current directions of interest include:

1. large-scale, ubiquitous sensor systems
2. bio-based or biomimetic systems
3. medical in situ applications
4. wireless sensor networks

Representative large-scale examples include intelligent vehicle and road systems, as well as instrumented contexts such as cities. Biological principles may provide fundamentally distinct approaches to the exploitation of dense, redundant, correlated, noisy sensors, especially when considered as part of a Gibbsian framework for behavioral response to environmen-

tal stimuli. Another issue here is the development of a theoretical understanding of sensor system development, adaptivity, and learning with respect to the particular context in which the system is deployed.

Further pushing the envelope of both technology and theory will permit the introduction of micro- and nanosensors into the human body and allow the monitoring and locally adaptive treatment of various illnesses. Finally, a more-complete theoretical framework that encompasses system models for wireless sensor networks is still required. This should include models of the physical phenomena being monitored, as well as operational and network issues. Finally, numerical analysis of the algorithmic properties of data-driven systems with sensor data error sources must be unified with the analysis of truncation, roundoff, and other errors.

A firm foundation exists upon which to build these new theories, systems, and applications. It will be a vibrant area of research for years to come.

References

- 25.1 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
- 25.2 J.O. Berger: *Statistical Decision Theory and Bayesian Analysis* (Springer, Berlin, Heidelberg 1985)
- 25.3 A. Elfes: Sonar-based real-world mapping and navigation, *IEEE Trans. Robot. Autom.* **3**(3), 249–265 (1987)
- 25.4 A. Elfes: Integration of sonar and stereo range data using a grid-based representation, *Proc. IEEE Int. Conf. Robot. Autom.* (1988) pp. 727–
- 25.5 L.D. Stone, C.A. Barlow, T.L. Corwin: *Bayesian Multiple Target Tracking* (Artech House, Norwood 1999)
- 25.6 Y. Bar-Shalom: *Multi-Target Multi-Sensor Tracking* (Artec House, Norwood 1990)
- 25.7 Y. Bar-Shalom, T.E. Fortmann: *Tracking and Data Association* (Academic, New York 1988)
- 25.8 P.S. Maybeck: *Stochastic Models, Estimation and Control*, Vol. I (Academic, New York 1979)
- 25.9 W. Sorensen: Special issue on the applications of the Kalman filter, *IEEE Trans. Autom. Control* **28**(3), 254–255 (1983)
- 25.10 B.D.O. Anderson, J.B. Moore: *Optimal Filtering* (Prentice Hall, Upper Saddle River 1979)
- 25.11 J. Manyika, H.F. Durrant-Whyte: *Data Fusion and Sensor Management: An Information-Theoretic Approach* (Prentice Hall, Upper Saddle River 1994)
- 25.12 S. Sukkarieh, E. Nettleton, J.H. Kim, M. Ridley, A. Goktogan, H. Durrant-Whyte: The ANSER project: Data fusion across multiple uninhabited air vehicles, *Int. J. Robot. Res.* **22**(7), 505–539 (2003)
- 25.13 R.E. Moore: *Interval Analysis* (Prentice Hall, Upper Saddle River 1966)
- 25.14 D. Dubois, H. Prade: *Fuzzy Sets and Systems: Theory and Applications* (Academic, New York 1980)
- 25.15 S. Blackman, R. Popoli: *Design and Analysis of Modern Tracking Systems* (Artec House, Norwood 1999)
- 25.16 D. Pagac, E.M. Nebot, H. Durrant-Whyte: An evidential approach to map-building for autonomous vehicles, *IEEE Trans. Robot. Autom.* **14**(4), 623–629 (1998)
- 25.17 D. Hall, J. Llinas: *Handbook of Multisensor Data Fusion* (CRC, Boca Raton 2001)
- 25.18 E.L. Waltz, J. Llinas: *Sensor Fusion* (Artec House, Norwood 1991)
- 25.19 M. Kam, Z. Zhu, P. Kalata: Sensor fusion for mobile robot navigation, *IEEE Proc.* **85**, 108–119 (1997)
- 25.20 H. Carvalho, W. Heinzelman, A. Murphy, C. Coelho: A general data fusion architecture, *Proc. 6th Int. Conf. Inf. Fusion*, Cairns (2003)
- 25.21 A. Makarenko: A Decentralized Architecture for Active Sensor Networks. Ph.D. Thesis (University of Sydney, Sydney 2004)

- 25.22 M. Dekhil, T. Henderson: Instrumented logical sensors systems, *Int. J. Robot. Res.* **17**(4), 402–417 (1998)
- 25.23 J.A. Profeta: Safety-critical systems built with COTS, *IEEE Comput.* **29**(11), 54–60 (1996)
- 25.24 H. Hu, J.M. Brady, F. Du, P. Probert: Distributed real-time control of a mobile robot, *J. Intell. Autom. Soft Comput.* **1**(1), 63–83 (1995)
- 25.25 S.A. Schneider, V. Chen, G. Pardo: ControlShell: A real-time software framework, *AIAA Conf. Intell. Robot. Field Fact. Serv. Space* (1994)
- 25.26 D. Simon, B. Espiau, E. Castillo, K. Kapellos: Computer-aided design of a generic robot controller handling reactivity and real-time issues, *IEEE Trans. Control Syst. Technol.* **4**(1), 213–229 (1993)
- 25.27 C. Giraud, B. Jouvencel: Sensor selection in a fusion process: a fuzzy approach, *Proc. IEEE Int. Conf. Multisens. Fusion Integr.*, Las Vegas (1994) pp. 599–606
- 25.28 R. Kapur, T.W. Williams, E.F. Miller: System testing and reliability techniques for avoiding failure, *IEEE Comput.* **29**(11), 28–30 (1996)
- 25.29 K.H. Kim, C. Subbaraman: Fault-tolerant real-time objects, *Commun. ACM* **40**(1), 75–82 (1997)
- 25.30 D.B. Stewart, P.K. Khosla: Mechanisms for detecting and handling timing errors, *Commun. ACM* **40**(1), 87–93 (1997)
- 25.31 G. Weller, F. Groen, L. Hertzberger: A sensor processing model incorporating error detection and recovery. In: *Traditional and Non-Traditional Robotic Sensors*, ed. by T. Henderson (Springer, Berlin, Heidelberg 1990) pp. 351–363
- 25.32 R.R. Brooks, S. Iyengar: *Averaging algorithm for multi-dimensional redundant sensor arrays: resolving sensor inconsistencies*, Tech. Rep. (Louisiana State University, Baton Rouge 1993)
- 25.33 T.C. Henderson, M. Dekhil: *Visual target based wall pose estimation*, Tech. Rep. UUCS-97-010 (University of Utah, Salt Lake City 1997)
- 25.34 S. Iyengar, D. Jayasimha, D. Nadig: A versatile architecture for the distributed sensor integration problem, *IEEE Comput.* **43**, 175–185 (1994)
- 25.35 D. Nadig, S. Iyengar, D. Jayasimha: New architecture for distributed sensor integration, *IEEE SOUTHEASTCON Proc.* (1993)
- 25.36 L. Prasad, S. Iyengar, R.L. Kashyap, R.N. Madan: Functional characterization of fault tolerant integration in distributed sensor networks, *IEEE Trans. Syst. Man Cybern.* **25**, 1082–1087 (1991)
- 25.37 L. Prasad, S. Iyengar, R. Rao, R. Kashyap: Fault-tolerance sensor integration using multiresolution decomposition, *Am. Phys. Soc.* **49**(4), 3452–3461 (1994)
- 25.38 H.F. Durrant-Whyte: *Integration, Coordination, and Control of Multi-Sensor Robot Systems* (Kluwer Academic, Boston 1987)
- 25.39 F. Groen, P. Antonissen, G. Weller: Model based robot vision, *IEEE Instrum. Meas. Technol. Conf.* (1993) pp. 584–588
- 25.40 R. Joshi, A.C. Sanderson: Model-based multisensor data fusion: a minimal representation approach, *Proc. IEEE Int. Conf. Robot. Autom.* (1994)
- 25.41 A.J. Briggs, B.R. Donald: Automatic sensor configuration for task-directed planning, *Proc. IEEE Int. Conf. Robot. Autom.* (1994) pp. 1345–1350
- 25.42 G. Hager: *Task Directed Sensor Fusion and Planning* (Kluwer Academic, Boston 1990)
- 25.43 G. Hager, M. Mintz: Computational methods for task-directed sensor data fusion and sensor planning, *Int. J. Robot. Res.* **10**(4), 285–313 (1991)
- 25.44 B. Donald: On information invariants in robotics, *Artif. Intell.* **72**, 217–304 (1995)
- 25.45 V. Braitenberg: *Vehicles: Experiments in Synthetic Psychology* (MIT Press, Cambridge 1984)
- 25.46 R.A. Brooks: A robust layered control system for a mobile robot, *IEEE Trans. Robot. Autom.* **2**(1), 14–23 (1986)
- 25.47 K.P. Valavanis, A.L. Nelson, L. Doitsidis, M. Long, R.R. Murphy: Validation of a distributed field robot architecture integrated with a matlab based control theoretic environment: A case study of fuzzy logic based robot navigation, *CRASAR 25* (University of South Florida, Tampa 2004)
- 25.48 R.R. Murphy: *Introduction to AI Robotics* (MIT Press, Cambridge 2000)
- 25.49 S. Lee: Sensor fusion and planning with perception-action network, *Proc. IEEE Conf. Multisens. Fusion Integr. Intell. Syst.*, Washington (1996)
- 25.50 S. Lee, S. Ro: Uncertainty self-management with perception net based geometric data fusion, *Proc. IEEE Conf. Robot. Autom.*, Albuquerque (1997)
- 25.51 B.A. Draper, A.R. Hanson, S. Buluswar, E.M. Riesenman: Information acquisition and fusion in the mobile perception laboratory, *Proc. SPIE – Signal Processing, Sensor Fusion, and Target Recognition VI*, Vol. 2059 (1996) pp. 175–187
- 25.52 S.S. Shafer, A. Stentz, C.E. Thorpe: An architecture for sensor fusion in a mobile robot, *Proc. IEEE Int. Conf. Robot. Autom.* (1986) pp. 2002–2007
- 25.53 S. Nagata, M. Sekiguchi, K. Asakawa: Mobile robot control by a structured hierarchical neural network, *IEEE Control Syst. Mag.* **10**(3), 69–76 (1990)
- 25.54 M. Pachter, P. Chandler: Challenges of autonomous control, *IEEE Control Syst. Mag.* **18**(4), 92–97 (1998)
- 25.55 R. Joshi, A.C. Sanderson: *Multisensor Fusion* (World Scientific, Singapore 1999)
- 25.56 V. Berge-Cherfaoui, B. Vachon: Dynamic configuration of mobile robot perceptual system, *Proc. IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems*, Las Vegas (1994)
- 25.57 V. Clement, G. Giraudon, S. Houzelle, F. Sandakly: Interpretation of remotely sensed images in a con-

- text of multisensor fusion using a multi-specialist architecture, Rapport de Recherche: Programme 4 – Robotique, Image et Vision 1768, INRIA (1992)
- 25.58 J. Albus: RCS: A cognitive architecture for intelligent multi-agent systems, Proc. IFAC Symp. Intell. Auton. Veh., Lisbon (2004)
- 25.59 R. Camden, B. Bodt, S. Schipani, J. Bornstein, R. Phelps, T. Runyon, F. French: *Autonomous Mobility Technology Assessment: Interim Report*, ARL-MR 565 (Army Research Laboratory, Washington 2003)
- 25.60 T. Queeney, E. Woods: A generic architecture for real-time multisensor fusion tracking algorithm development and evaluation, Proc. SPIE – Signal Processing, Sensor Fusion, and Target Recognition VII, Vol. 2355 (1994) pp. 33–42
- 25.61 T. Henderson, E. Shilcrat: Logical sensor systems, J. Robot. Syst. **1**(2), 169–193 (1984)
- 25.62 T. Henderson, C. Hansen, B. Bhanu: The specification of distributed sensing and control, J. Robot. Syst. **2**(4), 387–396 (1985)
- 25.63 J.D. Elliott: Multisensor fusion within an encapsulated logical device architecture. Master's Thesis (University of Waterloo, Waterloo 2001)
- 25.64 M.D. Naish: Elsa: An intelligent multisensor integration architecture for industrial grading tasks. Master's thesis (University of Western Ontario, London 1998)
- 25.65 A. Makarenko, A. Brooks, S. Williams, H. Durrant-Whyte, B. Grocholsky: A decentralized architecture for active sensor networks, Proc. IEEE Int. Conf. Robot. Autom., New Orleans (2004) pp.1097–1102
- 25.66 B. Grocholsky, A. Makarenko, H. Durrant-Whyte: Information-theoretic coordinated control of multiple sensor platforms, Proc. IEEE Int. Conf. Robot. Autom., Taipei (2003) pp.1521–1527
- 25.67 R. Gregor, M. Lützel, M. Pellkofer, K.-H. Siedersberger, E. Dickmanns: EMS-Vision: A perceptual system for autonomous vehicles, IEEE Trans. Intell. Transp. Syst. **3**(1), 48–59 (2002)
- 25.68 B. Rao, H. Durrant-Whyte, A. Sheen: A fully decentralized multi-sensor system for tracking and surveillance, Int. J. Robot. Res. **12**(1), 20–44 (1993)
- 25.69 B. Upcroft: Non-gaussian state estimation in an outdoor decentralised sensor network, Proc. IEEE Conf. Decis. Control (CDC) (2006)
- 25.70 S. Kumar, F. Ramos, B. Upcroft, H. Durrant-Whyte: A statistical framework for natural feature representation, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), Edmonton (2005) pp.1–6

Manipulation

Part D

Part D Manipulation and Interfaces

Ed. by Makoto Kaneko

26 Motion for Manipulation Tasks

Oliver Brock, Amherst, USA
James Kuffner, Pittsburgh, USA
Jing Xiao, Charlotte, USA

27 Contact Modeling and Manipulation

Imin Kao, Stony Brook, USA
Kevin Lynch, Evanston, USA
Joel W. Burdick, Pasadena, USA

28 Grasping

Domenico Prattichizzo, Siena, Italy
Jeffrey C. Trinkle, Troy, USA

29 Cooperative Manipulators

Fabrizio Caccavale, Potenza, Italy
Masaru Uchiyama, Sendai, Japan

30 Haptics

Blake Hannaford, Seattle, USA
Allison M. Okamura, Baltimore, USA

31 Telerobotics

Günter Niemeyer, Stanford, USA
Carsten Preusche, Wessling, Germany
Gerd Hirzinger, Wessling, Germany

32 Networked Telerobots

Dezhen Song, College Station, USA
Ken Goldberg, Berkeley, USA
Nak Young Chong, Ishikawa, Japan

33 Exoskeletons for Human Performance Augmentation

Homayoon Kazerooni, Berkeley, USA

Part D Manipulation and Interfaces, is separated into two subparts; the first half is concerned with manipulation where frameworks of modeling, motion planning, and control of grasp and manipulation of an object are addressed, and the second half is concerned with interfaces where physical human–robot interactions are handled. Humans can achieve grasping and manipulation of an object dexterously through hand–arm coordination. An optimum control skill for such a redundant system is naturally and gradually acquired through experience in our daily life. Especially, fingers play an important role for expressing human dexterity. Without dexterous fingers, it is hard for us to handle any daily tool, such as a pencil, keyboard, cup, knife, or fork. This dexterity is supported with active and passive compliance as well as the multiple sensory organs existing at the fingertip. Such dexterous manipulation enables us to clearly differentiate humans from other animals. Thus, manipulation is one of the most important functions for humans. We eventually acquired the current shape of finger, the sensory organs, and skill for manipulation, through a long history of evolution, over more than six million years. While both humans and robots are largely different in terms of actuators, sensors, and mechanisms, achieving dexterous manipulation like that of a human in a robot is a challenging subject in robotics. As we overview current robot technology, however, we observe that the dexterity of robots is still far behind that of humans. With this overview, we now provide a brief synopsis of each chapter in the first half of Part D.

Chapter 26, Motion for Manipulation Tasks, discusses algorithms that generate motion for manipulation tasks at the arm level, especially in an environment, by using the configuration space formalism. While in previous chapters (6 and 7) the focus was on specific algorithmic techniques for robot motion, this chapter is focused on a specific application for robot manipulation.

Chapter 27, Contact Modeling and Manipulation, provides contact modeling based on soft as well as rigid contacts. Kinematics and mechanics with friction are precisely handled under rigid-body contact. The selection matrix H is introduced to understand the force and velocity constraints at the contact interface. Pushing manipulation is also addressed by using the concept of the friction limit surface.

Chapter 28, Grasping, discusses, based on the closure property, grasping with many examples, supposing multifingered robotic hands. A strong constraint for grasping is the unilateral characteristic, where a fingertip can push but not pull an object through a contact point.

Kinematics and closure issues are addressed under this unilateral constraint.

Chapter 29, Cooperative Manipulators, addresses the strategies for controlling both the motion of cooperative system and the interaction forces between the manipulators and the grasped object when two manipulator arms firmly grasp a common object. It should be noted that this chapter allows the bilateral constraint where both directional force and moment are permissible.

Without dexterity like that of humans, future robots will not be able to work instead of humans in environments where human cannot enter. In this sense, the implementation of dexterity into robots is one of the highlights of future robot design. Chapters 26–29 provide a good hint for enhancing dexterity for robots.

The second half of Part D addresses interfaces where humans control a robot or multiple robots through direct or indirect contact with robot(s). We now provide a brief synopsis of each chapter in the second half of Part D.

Chapter 30, Haptics, discusses robotics devices that allow human operators to experience the sense of touch in remote or virtual environment. In haptic device design, two classes of haptic devices are discussed. One is an admittance haptic device that senses the force applied by the operator and constrains the operator's position to match the appropriate deflection of a simulated object or surface; the other is an impedance haptic device that senses the position of the operator and then applies a force vector to the operator according to the computed behavior of the simulated object or surface.

Chapter 31, Telerobotics, starts with a discussion on the classification of three different concepts: direct control where all slave motions are directly controlled by the user via the master interface, shared control where task execution is shared between direct control and local sensory control, and supervisory control where the user and slave are connected loosely with strong local autonomy. Various control issues such as time delay are also addressed.

Chapter 32, Networked Telerobotics, includes three components: a user, where anyone with an Internet connection is allowed, a web server on which an Internet-compatible server software can run, and a robot including a robot manipulator, mobile robot, or any device that can change its environment. The Internet is a powerful tool for controlling a robot (or robots), since it is the most popular communication media. We may see an enormous market for this in the future, and great care is required to ensure security.

Chapter 33, Exoskeletons for Human Performance Augmentation, deals with power-assist systems where human power is enhanced by wearing a robotic suit. In a human–robot system, we can consider that the distance between the human and the robot is zero as an extreme case. Design of hardware and control issues are precisely discussed by taking the Berkeley Lower Extremity Exoskeleton (BLEEX) as an example.

In both Telerobotics (Chap. 31) and Networked Telerobotics (Chap. 32), an appropriate distance between the human and robot is kept, while in both Haptics (Chap. 30) and Exoskeletons (Chap. 33), direct contact between human and robot is made. One of the main issues is how to maintain appropriate control performance of the system in the presence of humans or a time lag between humans and robots.

Motion for Manipulation Tasks

Oliver Brock, James Kuffner, Jing Xiao

This chapter serves as an introduction to Part D by giving an overview of motion generation and control strategies in the context of robotic manipulation tasks. Automatic control ranging from the abstract, high-level task specification down to fine-grained feedback at the task interface are considered. Some of the important issues include modeling of the interfaces between the robot and the environment at the different time scales of motion and incorporating sensing and feedback. Manipulation planning is introduced as an extension to the basic motion planning problem, which can be modeled as a hybrid system of continuous configuration spaces arising from the act of grasping and moving parts in the environment. The important example of assembly motion is discussed through the analysis of contact states and compliant motion control. Finally, methods aimed at integrating global planning with state feedback control are summarized.

26.1 Overview	615
26.2 Task-Level Control	618
26.2.1 Operational Space Control	618

26.2.2 Combined Force and Position Control	619
26.2.3 Operational Space Control of Redundant Mechanisms	619
26.2.4 Combining Mobility and Manipulation	620
26.2.5 Combining Multiple Task Behaviors	620
26.3 Manipulation Planning	622
26.3.1 Configuration Space Formalism	622
26.3.2 Example of a Three-DOF Planar Manipulator	625
26.3.3 Inverse Kinematics Considerations	626
26.4 Assembly Motion	628
26.4.1 Topological Contact States	629
26.4.2 Passive Compliance	629
26.4.3 Active Compliant Motion	631
26.5 Unifying Feedback Control and Planning	634
26.5.1 Feedback Motion Planning	635
26.5.2 Augmenting Global Plans with Feedback	636
26.6 Conclusions and Further Reading	637
References	639

26.1 Overview

Part D of this handbook is concerned with the interfaces that connect robots to their environment. We differentiate three such interfaces, depicted in Fig. 26.1. The first interface, between the robot and the computer, is primarily concerned with the automatic generation of motion for performing a task. The second interface is concerned with the physical interaction between the robot and the environment. This chapter relates both interfaces in the context of manipulation tasks. It is focused on automatic specification, planning, and execution of

motion of the robot manipulator or the manipulated object to meet a task goal under the constraints of the physical environment. Chapters 27–29 address other important issues pertaining to the second interface, such as the physical characteristics of contacts between the robot and the environment, grasping (i. e., the interaction between the robot and the object for manipulation), and the interaction between cooperative manipulators. The third interface, described in Chaps. 30–33, lies between humans and robots. Key issues that need to be

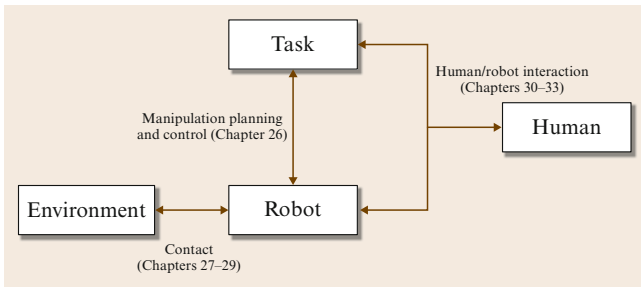


Fig. 26.1 Overview of Part D

addressed involve displaying appropriate sensor information to humans, or permitting humans to interactively specify the task or motion a robot is supposed to perform.

Manipulation refers to the process of moving or rearranging objects in the environment [26.1]. To perform a *manipulation task*, a robot establishes physical contact with objects in the environment and subsequently moves these objects by exerting forces and moments. The object to be manipulated can be large or small, and may serve various purposes in relation to the manipulation task (e.g., flipping a switch, opening a door, polishing a surface). In the context of automated assembly and industrial manipulation, the object to be manipulated is often referred to as a *part*. The *end-effector* generally refers to the link of the *manipulator* that makes contact and applies forces to the *part*. The *end-effector* is so named as it is often the end link in a serial kinematic chain. However, complex manipulation tasks may require the simultaneous exertion of multiple forces and moments at different contact points as well

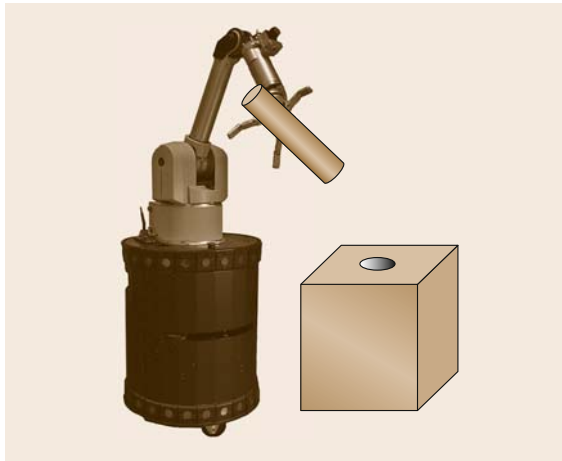


Fig. 26.2 A mobile manipulator inserting a peg into a hole

as the execution of sequences of force applications to several objects.

To understand the challenge of generating the motion required for the execution of a manipulation task, we will consider the classic example of inserting a peg into a hole. The mobile manipulator shown in Fig. 26.2 is commanded to perform the peg insertion *manipulation task*, whose general problem structure arises in a number of contexts such as compliant assembly. For simplicity, we will only consider the *transfer motion* (see Sect. 26.3) and assume that the robot has already established a stable grasp (see Chap. 28) and is holding the peg. If the clearance between the peg and the hole is very small, the peg may not be able to enter the hole easily, and often some contact occurs. In order to guide the peg successfully into the hole, the robot has to deal with the possible contact states between the peg and the hole (Sect. 26.4) and select an appropriate sequence of control strategies that will result in successful completion of the task.

The peg-and-hole example illustrates that successful robotic manipulation involves dealing with models of object geometry and contact in the presence of uncertainty. There are various kinds of uncertainty, including modeling, actuation, and sensing uncertainty. To plan motion strategies in the presence of uncertainty poses difficult computational challenges. Historically, motion planning for manipulation has been divided into *gross motion planning* and *fine motion planning*. The former involves reasoning about the manipulator motion on the macro scale and considers its overall global movement strategy, while the latter considers how to deal with uncertainty to accomplish a task requiring high precision robustly. The primary reason for the historical division stems from attempts to simplify the various aspects of the problem into something computationally tractable. Assuming that the uncertainty in actuator position, object models, and environment obstacle shapes can be bounded, gross motion planning can be formulated primarily in terms of geometry (Sect. 26.3). As the robot or the manipulated object makes contact with the environment, such as in the example of the insertion task, fine motion planning techniques are employed to take into account contact geometry, forces, friction, and uncertainty effectively (see Mason's book [26.1] and [26.2, 3] for an overview and historical perspective on fine motion strategies for manipulation tasks).

This peg-and-hole insertion example also illustrates that motion in the context of manipulation tasks is subject to *constraints*. These constraints have to be maintained to perform the task successfully. The specific

constraints depend on the type of manipulation task, but they may include contact constraints, position, and force constraints for the point at which the manipulator makes contact with the environment, kinematic and dynamic constraints imposed by the mechanism and its actuation capabilities, posture constraints that specify behavior to be performed in addition to the manipulation task, reactive obstacle avoidance in an unpredictably changing environment, and global motion constraints to ensure that a specific goal location is attained. These motion constraints are imposed by the task, by the kinematics of the mechanism performing the task, by the actuation capabilities of the mechanism, and by the environment.

To ensure that constraints are satisfied in spite of uncertainty, sensory information (see Part C – Sensing and Perception) is considered in the context of feedback control loops (Chaps. 6 and 35). Depending on the type of constraint, this feedback has to be considered at various time scales. For example, the exertion of a constant force on an object in the environment requires high-frequency feedback at rates up to 1000 Hz. At the other end of the time scale, we consider changes to the global connectivity of the environment. These changes, resulting from opening doors or moving obstacles, for example, occur relatively slowly or infrequently and feedback about these changes only has to be considered a few times per second. Figure 26.3 graphically illustrates a task-dependent ordering of motion constraints encountered in manipulation tasks and their associated feedback requirements.

This chapter is concerned with algorithms that generate motion for manipulation tasks. Previous chapters already discussed planning algorithms (Chap. 5) and control methods (Chap. 6). While in these two former chapters the focus was on specific algorithmic techniques for robot motion, the current chapter is focused on a specific application, namely robotic manipulation. This application dictates motion constraints and their feedback requirements that will have to be satisfied during the motion for a manipulation task. In contrast, the methods discussed in the two aforementioned chapters each only address a subset of the motion constraints, as indicated in Fig. 26.3.

This chapter first discusses task-level control in Sect. 26.2. Task-level control describes a set of techniques to control a robotic mechanism in terms of its contact points with the environment. These contact points are controlled to perform motion that accomplishes a specific manipulation task. This means that, instead of directly controlling the mechanism, task-

relevant points on the mechanism, so-called *operational points*, are controlled. This indirect control provides an intuitive means to specify desired motion for manipulation tasks.

Section 26.3 gives an overview of the configuration space formalization for manipulation planning, and how it can be cast as a classical motion planning problem (Chap. 5). Through this formalism, we can gain an understanding and intuition regarding how the geometry and relative positions of the manipulator, the manipulated objects, and the obstacles in the workspace determine both the geometry and topological structure of the manipulation configuration space. As we shall see, there are often an infinite number of possible ways to accomplish a manipulation task. Thus, the challenge in manipulation planning is to develop methods that deal effectively with the combinatorics of the search over all possible motions.

Section 26.4 addresses planning motion for assembly tasks, for which the peg-in-hole insertion is a typical example. The focus will be on motion constrained by contact, called *compliant motion*. There are two broad classes of assembly strategies involving compliant motion; one class is through special mechanism or control, called *passive compliance*, and the other is through active reasoning of contact states, called *active compliant motion*. Fine motion planning is discussed within the class of active compliant motion.

Methods from task-level control and manipulation planning address aspects of motion in a complementary fashion. While control methods are able to satisfy the feedback requirements of manipulation tasks, they often do not account for the overall global progress in accomplishing the manipulation task. Manipulation planners, on the other hand, reason about the global nature of the task at the high level, but are often computation-

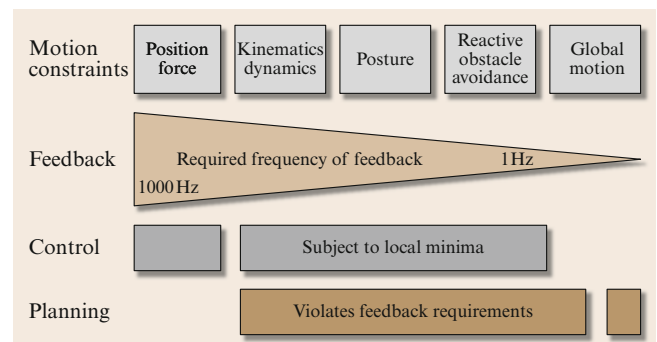


Fig. 26.3 Examples of motion constraints encountered in manipulation tasks and their feedback requirements

ally too expensive to deal with uncertainty or satisfy the high-frequency feedback requirements of manipulation tasks. A robust and skillful manipulator must satisfy all feedback requirements while at the same time en-

suring the accomplishment of the manipulation task. Section 26.5 discusses various techniques that offer a unified view of planning and control for manipulation tasks.

26.2 Task-Level Control

To perform a manipulation task, a robot has to establish contact with the environment. Through contact points the robot is able to apply forces and moments to objects. By controlling the position and velocity of the contact points as well as the forces acting at them, the robot causes the desired motion of objects, thereby performing the manipulation task. The programming of such a task is most conveniently accomplished by directly specifying positions, velocities, and forces at the contact points, rather than by specifying the joint positions and velocities required to achieve them.

Consider the manipulation task of serving a cup of water shown in Fig. 26.2. This task can easily be specified by providing a trajectory for the cup's motion. To determine a joint space trajectory that achieves the same motion, however, would be much more complex. One would have to rely on inverse kinematics (Sect. 1.9), which can be computationally challenging. More importantly, the task constraints imposed on the cup's motion do not uniquely specify the cup's trajectory. For example, while the cup is not allowed to tilt, it can be delivered to the goal location in any vertical orientation. Such task constraints can easily be specified in terms of the object motion – at the task level – but would be very difficult to characterize in terms of joint trajectories. Operational space control is therefore a natural choice for performing manipulation tasks. We will see in Sect. 26.2.3 that the operational space framework provides important additional advantages over joint space control in the context of redundant manipulators.

The advantages associated with task-level control come at the cost of depending on the manipulator Jacobian. For singular configurations of the manipulator (see Chap. 3), the manipulator Jacobian is not well defined. In these configurations, task-level control of a manipulator becomes unstable. Special care has to be applied to prevent the manipulator from entering these configurations, or to resort to specialized controllers in the proximity of these configurations. Most commonly, the manipulator is controlled so as to avoid singular configurations.

26.2.1 Operational Space Control

Task-level control – also called operational space control [26.4,5] (Sect. 6.2) – specifies the behavior of a robot in terms of operational points rather than joint positions and velocities. An operational point is an arbitrary point on the robot that is required to perform a particular motion or to exert a specified force to accomplish a manipulation task. In a serial-chain manipulator arm the operational point is most commonly chosen to coincide with the end-effector. More complex, branching mechanisms, such as humanoid robots, but also redundant serial-chain mechanisms, can have several operational points. To specify positions, velocities, accelerations, forces, and moments at the operational point, it is convenient to define a coordinate system with an origin that coincides with the operation point. This coordinate frame is called the operational frame. The orientation of the frame should be chosen in accordance with the task. For now we will ignore the orientation, however, and only consider tasks that require positioning of the operational point. In Sect. 26.2.2, we will consider more general tasks, combining position and force control.

Let us consider an operational point \mathbf{x} on a robot. The relationship between the joint velocities $\dot{\mathbf{q}}$ and the velocity of the operational point, $\dot{\mathbf{x}}$, is given by the Jacobian matrix of the mechanism (Sect. 1.8) at the operational point \mathbf{x} :

$$\dot{\mathbf{x}} = \mathbf{J}_{\mathbf{x}}(\mathbf{q})\dot{\mathbf{q}} . \quad (26.1)$$

Note that the Jacobian \mathbf{J} depends on the placement of the operational point \mathbf{x} , and on the current configuration \mathbf{q} of the robot. For simplicity, we will omit this dependency from our notation and just write $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$. From (26.1) we can derive an expression for the instantaneous torques $\boldsymbol{\tau}$ acting at the joints when the forces and moments described by \mathbf{F} is applied at the operational point \mathbf{x} :

$$\boldsymbol{\tau} = \mathbf{J}^T(\mathbf{q})\mathbf{F}_{\mathbf{x}} . \quad (26.2)$$

The vector \mathbf{F} captures the task to be performed at operational point \mathbf{x} . If the task specifies the complete position and orientation of the operational frame, \mathbf{F} will be given by $\mathbf{F} = (f_x, f_y, f_z, u_x, u_y, u_z)$, where f and u designate forces along and moments about the respective axes. (For easy of presentation, we will refer to \mathbf{F} as a force vector, even if it describes forces and moments.) The execution of such a task requires a manipulator with at least six degrees of freedom. If a task does not specify one of the components of \mathbf{F} , this component can be omitted and the corresponding column of the Jacobian matrix is dropped (or a manipulator with fewer than six degrees of freedom can be used to accomplish the task). For example, dropping u_z would indicate that the orientation about the z -axis is not specified by the task. The motion behavior in the dimensions of the task space that have been dropped is unspecified and the robot will float in these dimensions [26.4]. In Sect. 26.2.3 we will discuss how additional behavior can be performed in these unspecified dimensions of the task space.

Similarly to joint space control, we have to account for manipulator dynamics to achieve acceptable performance in operational space control. Using (26.1), we can project the joint space dynamics into operational space to obtain

$$\mathbf{F}_x = \mathbf{A}(\mathbf{q})\ddot{\mathbf{x}} + \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}), \quad (26.3)$$

where \mathbf{A} is the operational space inertial matrix, \mathbf{F} captures centrifugal and Coriolis forces in operational space, and $\boldsymbol{\eta}$ compensates for gravity forces. This equation pertains to a particular operational point. Intuitively speaking, the operational space inertia matrix \mathbf{A} captures the resistance of the operational point to acceleration along and about different axes. More details about operational space control and its relationship to joint space control are provided in Sect. 6.2 (motion control, joint space versus operational space control).

26.2.2 Combined Force and Position Control

Let us consider the example task of controlling the motion of a peg inside a hole. We assume the peg is already inserted into the hole and rigidly connected to the robot's end-effector. We attach the operational frame to a point on the peg so that its z -axis is aligned with the desired direction of motion of the peg inside the hole. To perform this task, the robot has to control the position of the peg along the z -axis of the operational frame and it also has to control contact forces between the peg and the hole to avoid jamming. Thus, the task of moving a peg

inside a hole requires that position and force control be seamlessly integrated.

To address contact forces within the operational space framework, we rewrite (26.3) as

$$\mathbf{F}_x = \mathbf{F}_c + \mathbf{A}(\mathbf{q})\mathbf{F}_m + \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}), \quad (26.4)$$

where \mathbf{F}_c represents the contact forces acting at the end-effector [26.4]. We can replace $\ddot{\mathbf{x}}$ with \mathbf{F}_m because $\ddot{\mathbf{x}}$ is acting on a dynamically decoupled system, i. e., a system that acts like a unit point mass. We can now control forces and motion in the operational frame by selecting the following control structure:

$$\mathbf{F}_x = \mathbf{F}_m + \mathbf{F}_c, \quad (26.5)$$

where

$$\mathbf{F}_m = \mathbf{A}(\mathbf{q})\boldsymbol{\Omega}\mathbf{F}'_m + \mathbf{F}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{x}} + \boldsymbol{\eta}(\mathbf{q}), \quad (26.6)$$

$$\mathbf{F}_c = \mathbf{A}(\mathbf{q})\bar{\boldsymbol{\Omega}}\mathbf{F}'_c, \quad (26.7)$$

where $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}}$ represent complementary task specification matrices [26.4] that determine along which directions the end-effector is position controlled and along which it is force controlled. By selecting $\boldsymbol{\Omega}$ appropriately, the combination of position and force control can be tailored to the task. In its simplest form, $\boldsymbol{\Omega}$ and $\bar{\boldsymbol{\Omega}} = \mathbf{I} - \boldsymbol{\Omega}$ are diagonal matrices. The i -th diagonal entry of $\boldsymbol{\Omega}$ is 1 if the i -th operational coordinate of the end-effector is position controlled and 0 if it is force controlled. In this simplest case, positions and forces are controlled along axes of the same coordinate frame. The concept of task specification matrix can be extended to include differently oriented coordinate frames for position and force control [26.4].

Once the force \mathbf{F}_x has been computed as given by (26.5), the corresponding joint torques used to control the robot are computed based on (26.1).

26.2.3 Operational Space Control of Redundant Mechanisms

The full expressiveness of task-level control comes to bear in the context of redundant manipulators. A manipulator is considered redundant with respect to the task it is performing if it has more degrees of freedom than required by the task. For example, the task of holding a cup of water only specifies two degrees of freedom, namely the two rotations about the axes spanning the horizontal plane – this task has two degrees of freedom. The mobile manipulator shown in Fig. 26.2 has ten degrees of freedom, leaving eight redundant degrees of freedom with respect to the task.

The operational space framework for task-level control of redundant manipulators decomposes the overall motion behavior into two components. The first component is given by the task, specified in terms of forces and moments, \mathbf{F}_{task} , acting at an operational point. This vector \mathbf{F} is translated into a joint torque based on (26.2): $\boldsymbol{\tau} = \mathbf{J}^T \mathbf{F}_{\text{task}}$. For a redundant manipulator, however, the torque vector $\boldsymbol{\tau}$ is not uniquely specified and we can select from a set of task-consistent torque vectors. The operational space framework performs this selection by considering a secondary task, the so-called posture behavior, making up the second component of the overall motion behavior. Posture behavior can be specified by an arbitrary torque vector $\boldsymbol{\tau}_{\text{posture}}$. To ensure that this additional torque does not affect the task behavior (\mathbf{F}_{task}), the torque $\boldsymbol{\tau}_{\text{posture}}$ is projected into the null space \mathbf{N} [26.6] of the task Jacobian \mathbf{J} . The null space of the Jacobian is the space orthogonal to the one spanned by \mathbf{J} and will be of rank $N_J - k$, where N_J is the number of degrees of freedom of the manipulator and $k = \text{rank}(\mathbf{J})$. The torque $\mathbf{N}^T \boldsymbol{\tau}_{\text{posture}}$ resulting from the null space projection is task consistent, i. e., it is guaranteed not to affect on the behavior of the operational point. Since the vector $\mathbf{N}^T \boldsymbol{\tau}_{\text{posture}}$ may not lie entirely within the null space of \mathbf{J} , however, the execution of the posture behavior cannot be guaranteed.

The operational space task ($\mathbf{J}^T \mathbf{F}_{\text{task}}$) and the posture behavior ($\mathbf{N}^T \boldsymbol{\tau}_{\text{posture}}$) are combined to obtain the general expression for the torque-level decomposition of the overall motion behavior:

$$\boldsymbol{\tau} = \mathbf{J}^T \mathbf{F}_{\text{task}} + \mathbf{N}^T \boldsymbol{\tau}_{\text{posture}} . \quad (26.8)$$

The null-space projection \mathbf{N} associated with the task Jacobian \mathbf{J} can be obtained by

$$\mathbf{N} = \mathbf{I} - \tilde{\mathbf{J}} \mathbf{J} , \quad (26.9)$$

where \mathbf{I} is the identity matrix, and $\tilde{\mathbf{J}}$ is the dynamically consistent generalized inverse of \mathbf{J} , given by

$$\tilde{\mathbf{J}} = \mathbf{H}^{-1} \mathbf{J}^T \boldsymbol{\Lambda} , \quad (26.10)$$

where \mathbf{H} is the joint space inertia matrix and $\boldsymbol{\Lambda}$ is the operational space inertia matrix for the operational point for which \mathbf{J} is defined. The use of this specific inverse to compute the null-space projection results in the selection of the task-consistent torque vector that minimizes kinetic energy.

26.2.4 Combining Mobility and Manipulation

The operational space framework does not distinguish between degrees of freedom used for mobility and those

used for manipulation. The end-effector-centric perspective of this approach considers all degrees of freedom to be in service to position and move the end-effector. An explicit coordination of manipulation and mobility for task-level control is not necessary.

A computationally efficient approach for coordinating manipulation and mobility was introduced in [26.7]. A similar, schema-based coordination of manipulation and mobility in the presence of dynamic obstacles was demonstrated in [26.8]. This approach is based on (26.2) and projects the forces that are derived from various schemas into the configuration space of the robot to determine its motion. Another coordination approach for manipulation and mobility considers a given end-effector path and generates a path of the base that maintains manipulability criteria [26.9]. This underlying representation of mobile manipulators can also be used to generate obstacle avoidance behavior of the base for a given end-effector path [26.10].

The coordination of a nonholonomic mobility platform and a cart-pushing task is addressed in [26.11]. In [26.12], an analysis of the task space for a mobile base with two manipulators is presented. Such a system with two manipulator arms can be viewed as a branching kinematic chain. Section 26.6 discusses extensions to the operational space framework to such branching mechanisms.

26.2.5 Combining Multiple Task Behaviors

The concept of decomposing a redundant robot's overall motion into task and posture behavior can be generalized to an arbitrary number of behaviors [26.13, 14]. Assume a set of n tasks T_i , where T_i has higher priority than T_j if $i < j$. Every task T_i is associated with a force vector \mathbf{F}_i and the corresponding joint torque $\boldsymbol{\tau}_i$. These tasks can be performed simultaneously by projecting the joint torque associated with each task into the combined null space of all tasks with higher priority, just as task and posture behavior were combined in the previous section. The null-space projection ensures that the execution of task i does not affect the execution of tasks with higher priority, given by the set $\text{prec}(i)$.

Given a task i , the joint torque consistent with the set of higher priority tasks $\text{prec}(i)$ can be obtained by projecting $\boldsymbol{\tau}_i$ into the combined null space of $\text{prec}(i)$,

$$\boldsymbol{\tau}_{i|\text{prec}(i)} = \mathbf{N}_{\text{prec}(i)}^T \boldsymbol{\tau}_i , \quad (26.11)$$

where the combined null space $\mathbf{N}_{\text{prec}(i)}^T$ is computed as

$$\mathbf{N}_{\text{prec}(n)} = \mathbf{N}_{n-1} \mathbf{N}_{n-2} \cdots \mathbf{N}_1 . \quad (26.12)$$

We say that $\tau_{i|\text{prec}(i)}$ is the torque for task i given the preceding tasks $\text{prec}(i)$. Applying this torque to the robot will not affect the execution of higher-priority tasks. The original torque vector τ_i can be computed based on (26.2).

Given the ability to compute a task torque that is consistent with preceding tasks we are able to combine an arbitrary number of tasks into a single motion behavior:

$$\tau = \tau_1 + \tau_{2|\text{prec}(2)} + \tau_{3|\text{prec}(3)} + \cdots + \tau_{n|\text{prec}(n)}, \quad (26.13)$$

where τ is the torque used to control the robot.

Note that τ_1 is not projected into any null space, since it is the task with the highest priority. Task 1 is therefore performed in the full space defined by the robot's kinematics. If the robot has N degrees of freedom, this space has N dimensions. As the number of tasks performed on the robot increases, the null-space projections $N_{\text{prec}(i)}$ will project the N -dimensional torque vectors associated with lower-priority tasks into a subspace of decreasing dimension. Eventually, the null space will be reduced to the trivial case, when all torque vectors are projected onto the null vector. This prevents lower-priority tasks from being executed. The dimensionality of the null space of tasks $\text{prec}(i)$ can be computed by counting the nonzero singular values [26.6] of $N_{\text{prec}(i)}$, permitting a programmatic assessment of task feasibility.

If the execution of a particular task has to be guaranteed, it should be associated with task 1 in (26.13). Therefore, the highest-priority task is generally used for hard constraints that under no circumstances should be violated. These constraints may include contact constraints, joint limits, and balancing in the case of humanoid robots, for example (Chap. 56).

The projection of (26.11) maps the torque τ_i of task i into the null space of all preceding tasks. This projection also scales the vector, which may significantly impair the execution of task i . This effect can be reduced by scaling the desired accelerations \ddot{x} at the operational point with the task-consistent inertia matrix $\Lambda_{i|\text{prec}(i)}$, given by

$$\Lambda_{i|\text{prec}(i)} = \left(J_{i|\text{prec}(i)} H^{-1} J_{i|\text{prec}(i)}^\top \right)^{-1}, \quad (26.14)$$

where H is the joint space inertia matrix and

$$J_{i|\text{prec}(i)} = J_i N_{\text{prec}(i)} \quad (26.15)$$

is the task Jacobian consistent with all preceding tasks. This Jacobian projects operational space forces into joint torques that do not cause any acceleration at the operational points of tasks in $\text{prec}(i)$. The task-consistent

inertia matrix $\Lambda_{i|\text{prec}(i)}$ then captures the inertia perceived at the operational point when its motion is limited to the subspace consistent with all preceding tasks.

Using the task-consistent inertia matrix, we can obtain the task-consistent operational space dynamics by projecting the joint space dynamics into the space defined by the dynamically consistent inverse of $J_{i|\text{prec}(i)}$. Inverting (26.2) and replacing τ based on (6.1) we obtain

$$\bar{J}_{i|\text{prec}(i)}^\top (H(q) + C(q, \dot{q}) + \tau_g(q) = \tau_{k|\text{prec}(i)}), \quad (26.16)$$

which yields the task-consistent equivalent to (26.3):

$$F_{i|\text{prec}(i)} = \Lambda_{i|\text{prec}(i)} \ddot{x} + F_{i|\text{prec}(i)} \dot{x} + \eta_{i|\text{prec}(i)}, \quad (26.17)$$

where $\Lambda_{i|\text{prec}(i)}$ is the task-consistent operational space inertial matrix, $F_{i|\text{prec}(i)}$ captures centrifugal and Coriolis forces in operational space, and $\eta_{i|\text{prec}(i)}$ compensates for gravity forces. Equation (26.17) can be used to control a task at a particular operational point in a task-consistent manner, i.e., without causing acceleration at the operational points associated with the tasks in $\text{prec}(i)$. Note that in practice the terms $C(q, \dot{q})$ and $\tau_g(q)$ can be computed in the joint space; they only need to be accounted for once in the case of multiple tasks.

Referring back to Fig. 26.3, we can analyze the effectiveness of the operational space framework. The control of operational points directly addresses force and position constraints on motion in manipulation tasks. Additional behaviors with lower priority than the task can be used to ensure the maintenance of kinematic and dynamic constraints, implement posture behavior, or perform reactive obstacle avoidance based on artificial potential fields [26.15]. The control of all of these aspects of manipulation tasks is computationally efficient. As a consequence, the operational space framework is able to maintain these motion constraints while satisfying feedback frequency requirements. However, the operational space framework does not include the consideration of global motion constraints, such as those imposed by the connectivity of the work space. Therefore, the operational space framework is susceptible to local minima, similarly to the artificial potential field method. Such local minima can occur for all of the aforementioned motion constraints and would prevent the robot from accomplishing its task. To overcome this problem, planning methods have to be applied (see Chap. 5 and Sect. 26.3).

The control basis approach [26.16] provides an alternative, compact formalism for combining multiple

behaviors. Each individual behavior is described by a controller, represented by a potential function $\phi \in \Phi$, sensors $\sigma \in \Sigma$, and effectors $\tau \in \Upsilon$. A particular controller ϕ can be parameterized with sensors and effectors. A particular instance of a behavior can be written as $\phi|_{\tau}^{\sigma}$. These controllers can be combined using null-space projections, similar to (26.13). In the control basis framework, (26.13) would be written more compactly as

$$\phi_n \triangleleft \cdots \triangleleft \phi_3 \triangleleft \phi_2 \triangleleft \phi_1, \quad (26.18)$$

where we abbreviate $\phi_i|_{\tau_i}^{\sigma_i}$ with ϕ_i . The relation $\phi_i \triangleleft \phi_j$ means that ϕ_i is projected into the null space of ϕ_j . We say that ϕ_i is executed *subject to* ϕ_j . If more than

two behaviors are combined, as shown in (26.18), each controller ϕ_i is projected into the combined null space of all superior tasks.

The control basis represents a general framework for combining controllers and is not limited to the notion of task-level control. In addition to the aforementioned formalism for representing combinations of controllers, the control basis also encompasses discrete structures that define transitions between combinations of controllers. Such transitions can be specified by the user or learned with techniques from reinforcement learning [26.16]. The control basis can compose multiple task-level controllers to implement manipulation tasks that cannot be described in terms of a single task-level controller.

26.3 Manipulation Planning

In order to appreciate the computational challenges inherent to gross motion planning for manipulation, it is useful to adopt a mathematical formalism for analysis. The chapter on planning (Chap. 5) introduced the *configuration space* (*C-space*), which represents the space of allowable transformations of the robot. In general, the set of all allowable transformations forms an n -dimensional *manifold*, in which n is the total number of degrees of freedom (DOF) of the mechanical system. For a robotic manipulator, the configuration space is defined by its kinematic structure (see Chap. 1). One common class of manipulators is *serial-chain* manipulators, which consist of a series of revolute or prismatic joints connected to form a linear kinematic chain. The discussion of manipulation planning in this section focuses on configuration spaces arising from single serial-chain manipulators. However, the mathematical formulation generalizes to more complex robots such as humanoids (Fig. 26.4; Chap. 56), whose upper

body consists of two cooperating manipulator arms attached to a movable base (see Chap. 29). The model considered here greatly simplifies the problems of grasping, stability, friction, mechanics, and uncertainties and instead focuses on the geometric aspects. In this section, we will consider a simple example of a manipulation task that involves a classic pick-and-place operation. Section 26.4 further considers grasping and contact states in the context of assembly operations, and Sect. 26.5 gives an overview of techniques to unify feedback control and planning for more complex manipulation tasks.

26.3.1 Configuration Space Formalism

The basic motion planning problem is defined as finding a path in the configuration space connecting a start and a goal configuration while avoiding static obstacles in the environment (see Chap. 5). Manipulation planning involves additional complexity because manipulation tasks fundamentally require contact with objects in the environment. When objects are grasped, regrasped, and moved, the structure and topology of the free configuration space can change drastically. However, manipulation planning shares the same basic theoretical foundations and computational complexity as classical path planning, such as PSPACE-hardness (see Chap. 5).

To be more precise, we will utilize a formalization for manipulation planning that is primarily due to Alami et al. [26.17], and adapted in LaValle's book [26.18]. Our goal is to gain understanding and intuition regarding



Fig. 26.4 Simulation of the H6 humanoid robot manipulating an object

the search space for manipulation tasks and the different situations that can arise. Throughout this discussion, bear in mind that the geometry and relative positions of the manipulator, the manipulated objects, and the obstacles in the workspace determine both the geometry and topological structure of the free configuration space. Within this framework, manipulation planning can be intuitively placed in the context of planning for *hybrid systems* [26.19]. Hybrid systems involve a mixture of continuous variables and discrete *modes*, whose transitions are defined by switching functions (for a discussion of hybrid systems in a planning context, see Chap. 7 of LaValle's book [26.18]). In the case of manipulation planning, the effect of grasping, repositioning, and releasing objects in the environment corresponds to switching between different continuous configuration spaces.

Let us consider a simple example of a simple *pick-and-place manipulation task* that requires a robot manipulator \mathcal{A} to reposition a single movable rigid object (part) \mathcal{P} from its current location in the workspace to some desired goal location. Note that accomplishing the task only requires that the part \mathcal{P} reach the goal location, without regard to precisely *how* the manipulator does so. Because the part cannot move by itself, it must either be transported by the robot or be placed at rest in some stable intermediate configuration. There are some configurations in which the robot is able to grasp and move the part to other locations in the environment. There are also forbidden configurations in which the robot or the part collides with obstacles in the environment. The solution of a manipulation planning problem consists of a sequence of subpaths for the robot moving alone or moving while grasping the part. The primary constraints on the allowable motions are:

1. The robot manipulator must not collide with any obstacles in the environment while reaching for, grasping, or regrasping the part.
2. The robot must adequately grasp the part while transporting it (for details on grasping and metrics for stable grasps see Chap. 28).
3. The robot manipulator and movable part must not collide with any obstacles in the environment while the part is being transported.
4. If the part is not being transported by the robot, it must be placed at rest in some stable intermediate placement.
5. The part must end up at the desired goal location in the workspace.

The two modes of operation give rise to the hybrid systems formulation: either the robot moves alone, or the robot moves while grasping the part. In the literature, motions of the robot holding the object at a fixed grasp are called *transfer paths*, and motions of the robot while the part stays at a stable placement are called *transit paths* [26.17, 20, 21]. Solving a manipulation planning problem involves searching for a connected sequence of *transit* and *transfer* paths for the manipulator that will accomplish the task. For an autonomous robot, the ideal is to utilize planning algorithms that will automatically produce correct sequences of transfer and transit paths separated by grasping and ungrasping operations.

Admissible Configurations

Building on the notation introduced in Chap. 5, we define a robot manipulator \mathcal{A} with n degrees of freedom operating in a Euclidean *world* or *workspace*, $\mathcal{W} = \mathbb{R}^N$, in which $N = 2$ or $N = 3$. Let $\mathcal{C}^{\mathcal{A}}$ represent the n -dimensional \mathcal{C} -space of \mathcal{A} , and $q^{\mathcal{A}}$ be a configuration. $\mathcal{C}^{\mathcal{A}}$ is called the *manipulator configuration space*. Let \mathcal{P} denote a movable part, which is a solid rigid body modeled with geometric primitives. We will assume that \mathcal{P} is allowed to undergo rigid-body transformations. This gives rise to a *part configuration space*, $\mathcal{C}^{\mathcal{P}} = SE(2)$ or $\mathcal{C}^{\mathcal{P}} = SE(3)$. Let $q^{\mathcal{P}} \in \mathcal{C}^{\mathcal{P}}$ denote a part configuration. The area or volume in the workspace occupied by the transformed part model is denoted by $\mathcal{P}(q^{\mathcal{P}})$.

We define the combined (robot and part) configuration space \mathcal{C} as the Cartesian product

$$\mathcal{C} = \mathcal{C}^{\mathcal{A}} \times \mathcal{C}^{\mathcal{P}},$$

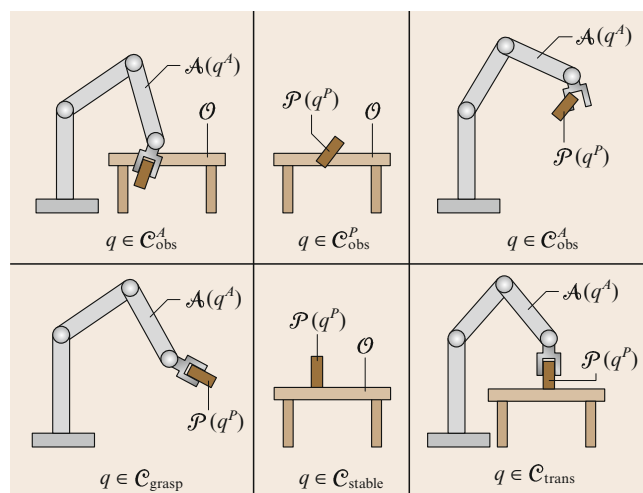


Fig. 26.5 Examples of the various configuration space subsets defined for manipulation planning

in which each configuration $q \in \mathcal{C}$ is of the form $q = (q^A, q^P)$. Note that the set of admissible configurations for manipulation planning purposes is more restrictive than simply $\mathcal{C}_{\text{free}}$, the space free of collisions with obstacle regions as defined in Chap. 5. We must incorporate the constraints on the allowable motions by removing inadmissible configurations. Figure 26.5 illustrates examples of some of the important subsets of \mathcal{C} for manipulation planning.

We start by removing all configurations that involve collisions with obstacles. Let the subset of configurations in which the manipulator collides with obstacles be given by

$$\mathcal{C}_{\text{obs}}^A = \{(q^A, q^P) \in \mathcal{C} \mid \mathcal{A}(q^A) \cap \mathcal{O} \neq \emptyset\}.$$

We also want to remove configurations for which the part collides with obstacles. However, we will allow the surface of the part to make contact with obstacle surfaces. This happens for example, when the part is resting on a shelf or table, or when a peg-shaped part is being inserted into a hole. Therefore, let

$$\mathcal{C}_{\text{obs}}^P = \{(q^A, q^P) \in \mathcal{C} \mid \text{int}(\mathcal{P}(q^P)) \cap \mathcal{O} \neq \emptyset\}$$

denote the open set for which \mathcal{O} intersects the interior volume $\text{int}(\mathcal{P}(q^P))$ of the part for any configuration q^P . If the interior of the part penetrates \mathcal{O} , then clearly these configurations should be avoided.

We can now define $\mathcal{C} \setminus (\mathcal{C}_{\text{obs}}^A \cup \mathcal{C}_{\text{obs}}^P)$, which is the set of all configurations in which both the robot and the part do not inappropriately collide with \mathcal{O} . Next consider the interaction between \mathcal{A} and \mathcal{P} . The manipulator must be allowed to make contact with the surface of the part, but penetration is once again not allowed. Therefore, let all configurations in which the interior of the part overlaps with the robot geometry be defined as

$$\mathcal{C}_{\text{obs}}^{PA} = \{(q^A, q^P) \in \mathcal{C} \mid \mathcal{A}(q^A) \cap \text{int}(\mathcal{P}(q^P)) \neq \emptyset\}.$$

Finally, we are able to define

$$\mathcal{C}_{\text{adm}} = \mathcal{C} \setminus (\mathcal{C}_{\text{obs}}^A \cup \mathcal{C}_{\text{obs}}^P \cup \mathcal{C}_{\text{obs}}^{PA}),$$

which represents the set of configurations remaining after subtracting away all unwanted configurations. We call this the set of *admissible configurations*.

Stable and Grasped Configurations

By definition, all configurations in the set \mathcal{C}_{adm} are free of penetrative collisions between the robot, the part, and obstacles in the environment. However, many of these configurations define the part as floating in space or on

the verge of falling. We now consider two important subsets of \mathcal{C}_{adm} that are used in manipulation planning (see Fig. 26.5). Let $\mathcal{C}_{\text{stable}}^P \subseteq \mathcal{C}^P$ denote the subset of *stable part configurations*, which are configurations at which the part can safely rest without any forces being applied by the manipulator. Examples of stable configurations include the part resting on a table or inserted into a larger assembly of other parts. The criteria for inclusion into the set of stable configurations depends on properties such as the part geometry, friction, mass distribution, and contacts with the environment. Our formalism does not consider these issues directly, but assumes that some method for evaluating the stability of a part configuration is available. Given $\mathcal{C}_{\text{stable}}^P$, we define $\mathcal{C}_{\text{stable}} \subseteq \mathcal{C}_{\text{adm}}$ to be the corresponding stable configurations of the part–robot system:

$$\mathcal{C}_{\text{stable}} = \{(q^A, q^P) \in \mathcal{C}_{\text{adm}} \mid q^P \in \mathcal{C}_{\text{stable}}^P\}.$$

The other important subset of \mathcal{C}_{adm} is the set of all configurations in which the robot is grasping the part and is capable of manipulating it according to some defined criteria. Let $\mathcal{C}_{\text{grasp}} \subseteq \mathcal{C}_{\text{adm}}$ denote the set of *grasp configurations*. For every configuration, $(q^A, q^P) \in \mathcal{C}_{\text{grasp}}$, the manipulator touches the part, which implies that $\mathcal{A}(q^A) \cap \mathcal{P}(q^P) \neq \emptyset$. Just as before, penetration between the robot and part geometry is not allowed because $\mathcal{C}_{\text{grasp}} \subseteq \mathcal{C}_{\text{adm}}$. In general, many configurations at which $\mathcal{A}(q^A)$ contacts $\mathcal{P}(q^P)$ will not necessarily be in $\mathcal{C}_{\text{grasp}}$. The criteria for a configuration to lie in $\mathcal{C}_{\text{grasp}}$ depends on the particular characteristics of the manipulator, the part, and the contact surfaces between them. For example, a typical manipulator would not be able to pick up a part by making contact at only a single point. (For additional information, see Sect. 26.4 for contact state identification, Chap. 27, and grasping criteria, such as *force closure* models in Chaps. 28 and 15).

For any robot and part configuration $q = (q^A, q^P) \in \mathcal{C}$ that we consider for manipulation planning we must always ensure that either $q \in \mathcal{C}_{\text{stable}}$ or $q \in \mathcal{C}_{\text{grasp}}$. We therefore define $\mathcal{C}_{\text{free}} = \mathcal{C}_{\text{stable}} \cup \mathcal{C}_{\text{grasp}}$, to reflect the subset of \mathcal{C}_{adm} that is permissible for manipulation planning. In the context of hybrid systems, this gives rise to the two distinct *modes* for manipulation planning: the *transit mode* and the *transfer mode*. Recall that in the transit mode, the manipulator is not carrying the part (i.e., only the robot moves), which requires that $q \in \mathcal{C}_{\text{stable}}$. In the transfer mode, the manipulator carries the part (i.e., both the robot and the part move), which requires that $q \in \mathcal{C}_{\text{grasp}}$. Based on these conditions, the only way the mode can change is if $q \in \mathcal{C}_{\text{stable}} \cap \mathcal{C}_{\text{grasp}}$. When the manipulator is in these

configurations, it has two available actions: (1) continue to grasp and move the part, or (2) release the part in its currently stable configuration. In all other configurations the mode remains unchanged. For convenience, we define $\mathcal{C}_{\text{trans}} = \mathcal{C}_{\text{stable}} \cap \mathcal{C}_{\text{grasp}}$ to denote the set of *transition configurations*, which are the places in which the mode may change and the robot may grasp or release the part.

Manipulation Planning Task Definition

Finally, the basic pick-and-place manipulation planning task can now be defined. An *initial part configuration*, $q_{\text{init}}^P \in \mathcal{C}_{\text{stable}}^P$, and a *goal part configuration*, $q_{\text{goal}}^P \in \mathcal{C}_{\text{stable}}^P$, are specified. Recall that this high-level task specification is only defined in terms of repositioning the part, without regard to how the manipulator actually accomplishes the task. Let $\mathcal{C}_{\text{init}} \subseteq \mathcal{C}_{\text{free}}$ be the set of all configurations in which the part configuration is q_{init}^P :

$$\mathcal{C}_{\text{init}} = \{(q^A, q^P) \in \mathcal{C}_{\text{free}} | q^P = q_{\text{init}}^P\}.$$

We define $\mathcal{C}_{\text{goal}} \subseteq \mathcal{C}_{\text{free}}$ similarly as the set of all configurations in which the part configuration is q_{goal}^P :

$$\mathcal{C}_{\text{goal}} = \{(q^A, q^P) \in \mathcal{C}_{\text{free}} | q^P = q_{\text{goal}}^P\}.$$

If the initial and final configurations of the manipulator are specified, then let $q_{\text{init}} = (q_{\text{init}}^A, q_{\text{init}}^P) \in \mathcal{C}_{\text{init}}$ and $q_{\text{goal}} = (q_{\text{goal}}^A, q_{text{goal}}^P) \in \mathcal{C}_{\text{goal}}$. The objective of the planner is to compute a path τ such that

$$\begin{aligned} \tau : [0, 1] &\mapsto \mathcal{C}_{\text{free}}, \\ \tau(0) &= q_{\text{init}}; \tau(1) = q_{\text{goal}}. \end{aligned}$$

If the initial and final positions of the manipulator are unspecified, we implicitly define the objective of the

planner as computing a path $\tau : [0, 1] \mapsto \mathcal{C}_{\text{free}}$ such that $\tau(0) \in \mathcal{C}_{\text{init}}$ and $\tau(1) \in \mathcal{C}_{\text{goal}}$. In either case, a solution is an alternating sequence of transit paths and transfer paths, whose names follow from the mode. In between each transfer path, the part is placed in a stable intermediate configuration while the manipulator moves alone (transit path) in order to regrasp the part. This sequence continues until the part is ultimately placed to rest in its final goal configuration. This is depicted in Fig. 26.6.

26.3.2 Example of a Three-DOF Planar Manipulator

For manipulators with three or fewer degrees of freedom (DOF), we can construct visualizations of the \mathcal{C} -space in order to gain an intuition of the structure. Throughout this section, we will use the example of a three-DOF serial-chain manipulator whose end-effector operates a planar workspace (\mathbb{R}^2).

Figure 26.7 shows a simple redundant robot arm originally illustrated in [26.22]. The robot has three revolute joints with parallel axes and joint limits of $-\pi, \pi$ radians, providing three degrees of freedom. Since the end-effector of the robot can only reach positions in a plane, the arm is redundant in this plane and since it has only three DOF and limits imposed on the joint angles, \mathcal{C} can be visualized as a cube with an edge length of 2π . Generally, the \mathcal{C} -space manifold of a manipulator with n revolute joints is homeomorphic to an n -dimensional *hypertorus* if no joint limits are present, and homeomorphic to an n -dimensional *hypercube* if joint limits are present.

Recall that obstacles in the workspace are mapped to regions in the \mathcal{C} -space called \mathcal{C} -obstacles (Chap. 5), which represent the set of all joint configurations of the robot that cause an intersection of the geometry of the robot and the obstacle

$$\mathcal{C}_{\text{obs}}^A = \{(q^A, q^P) \in \mathcal{C} | A(q^A) \cap \mathcal{O} \neq \emptyset\}.$$

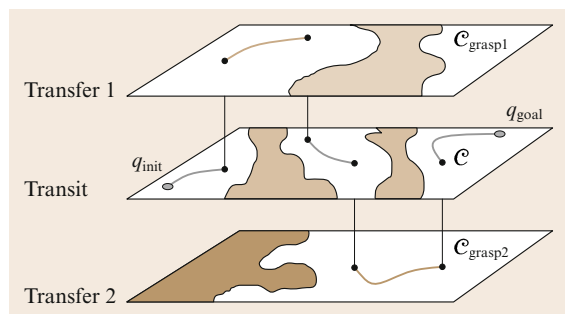


Fig. 26.6 Manipulation planning involves searching for a sequence of transfer and transit paths in a set of hybrid continuous configuration spaces. In this example, transfer paths in different \mathcal{C} -spaces arise from different ways of rigidly grasping a part

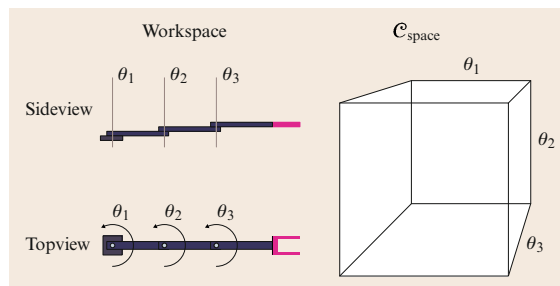


Fig. 26.7 Planar 3 DOF manipulator

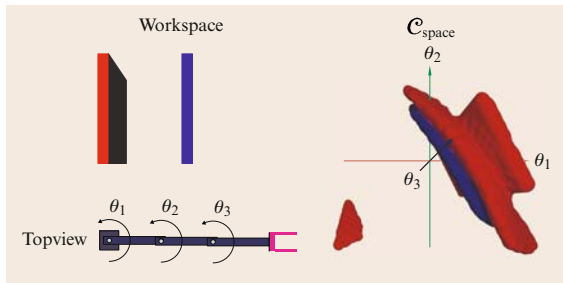


Fig. 26.8 Planar 3 DOF manipulator in a workspace with box-shaped obstacles and associated configuration space obstacles

In general, the structure of $\mathcal{C}_{\text{obs}}^A$ can be highly complex and involve multiple connected components of \mathcal{C} . An example visualization of a configuration space for the planar three-DOF manipulator corresponding to a workspace with two differently colored box-shaped obstacles is given in Fig. 26.8. The \mathcal{C} -obstacles are illustrated using the same color as the corresponding box-shaped obstacle in the workspace. The red, green, and blue coordinate axes correspond to the three joint angles $\theta_{1,2,3}$ of the manipulator, respectively.

Self-collisions induce special \mathcal{C} -obstacle regions that do not involve environment geometry, but only the geometric interference of the links of the robot itself. The robot shown in Fig. 26.7 is, by design, self-collision-free. Figure 26.9 shows a modified version of the manipulator, where self-collision is actually possible, and a visualization of the corresponding \mathcal{C} -obstacles.

\mathcal{C} -obstacles can cover large parts of the configuration space and even cause a complete disconnection between different components of $\mathcal{C}_{\text{free}}$. Figure 26.10 shows how the addition of only a small post as an obstacle can cause a wide range of configurations to produce collisions and disconnect regions of $\mathcal{C}_{\text{free}}$: the \mathcal{C} -obstacle corresponding to the post forms a *wall* of

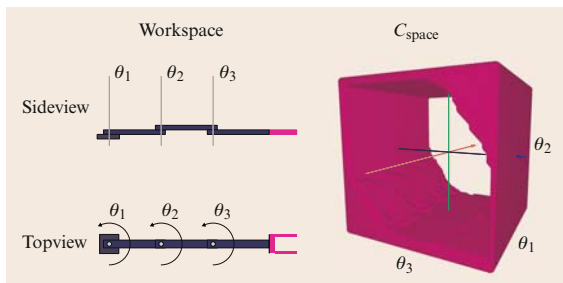


Fig. 26.9 Non-planar 3 DOF manipulator and corresponding self-collision \mathcal{C} -obstacle

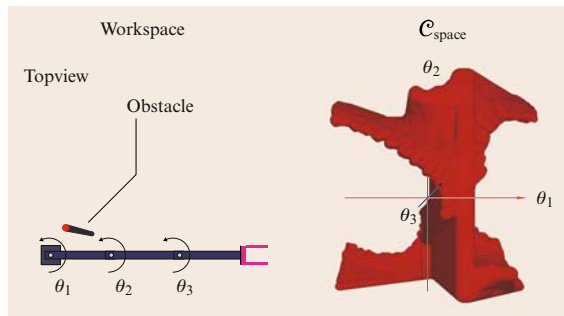


Fig. 26.10 Workspace mapping to a configuration space with topologically disconnected components

configurations parallel to the θ_2 – θ_3 -plane. This wall represents the range of values for θ_1 for which the first link of the manipulator intersects with the post, splitting $\mathcal{C}_{\text{free}}$ into two disconnected components. Having disconnected components $\mathcal{C}_1, \mathcal{C}_2 \in \mathcal{C}_{\text{free}}$ means that there is no collision-free path that starts in \mathcal{C}_1 and ends in \mathcal{C}_2 .

26.3.3 Inverse Kinematics Considerations

The typical formulation of a path planning problem requires a goal configuration q_{goal} as input (see Chap. 5). For the case of manipulation planning, this configuration is usually computed from the desired workspace pose of the end-effector by an inverse kinematics (IK) solver (see Sect. 1.7). Apart from special cases, there currently exist no known analytical methods for solving the inverse kinematics of a general redundant mechanism (greater than six degrees of freedom). Iterative, numerical techniques (such as Jacobian-based methods or gradient-based optimization) are typically used to calculate solutions in this case. Note that the numerical computation of inverse kinematics solutions can suffer from poor performance or even nonconvergence, depending on the quality of the initial guess, the distribution of singularities in the \mathcal{C} -space or a combination of these effects (see Sect. 1.7).

Most IK solvers are only able to compute one configuration for a given end-effector pose. For a redundant manipulator, however, an infinite, continuous range of configurations that cause the end-effector to assume the desired pose usually exists. In the context of path planning, this means that the IK solver will select one out of an infinite number of configurations as q_{goal} . Note that in some cases, the selected q_{goal} might not be collision-free, i. e., it is not part of $\mathcal{C}_{\text{free}}$. Such a configuration cannot be used as a goal for computing a global reaching strategy

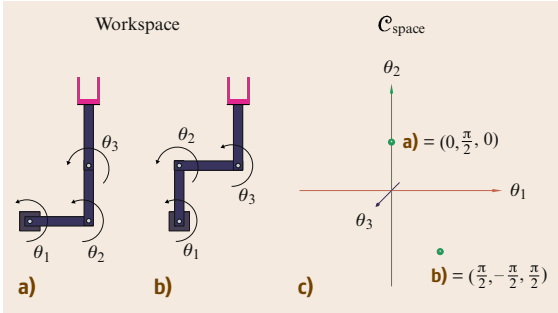


Fig. 26.11 Two inverse kinematics solutions for the same end-effector pose of a planar 3 DOF arm

because no solution exists for the corresponding path planning query in the classical sense (Chap. 5).

Another possibility is that the selected q_{goal} might be disconnected from q_{init} . In this case, q_{goal} is unsuitable for planning despite the fact that it is a member of $\mathcal{C}_{\text{free}}$, because it is in a component of $\mathcal{C}_{\text{free}}$ that is disconnected from the component in which q_{init} lies. Consider the planar manipulator shown in Fig. 26.7. Figure 26.11 shows two possible solutions for a certain end-effector pose. The corresponding configurations $q_a = (0, \frac{\pi}{2}, 0)$ and $q_b = (\frac{\pi}{2}, -\frac{\pi}{2}, \frac{\pi}{2})$ are marked by green spheres in the C-space visualization. In this case, the IK solver might select one of the two solutions as q_{goal} . If the same obstacle as in Fig. 26.10 is added to the workspace, the two solutions will lie in disconnected components of $\mathcal{C}_{\text{free}}$, as shown in Fig. 26.12.

Let $q_{\text{init}} = (0, 0, 0)$ be the zero configuration, marked by the point of intersection of the C-space axes in Fig. 26.12. Let the component of $\mathcal{C}_{\text{free}}$ in which it lies be denoted by \mathcal{C}_1 and the other component by \mathcal{C}_2 . Since $q_b \in \mathcal{C}_2$ and $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$, no solution for the planning problem can exist if the IK solver selects q_b as q_{goal} .

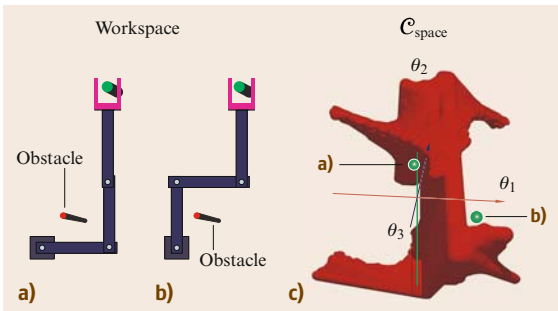


Fig. 26.12 Two inverse kinematics solutions lying in disconnected components of $\mathcal{C}_{\text{free}}$ due to a small obstacle in the workspace

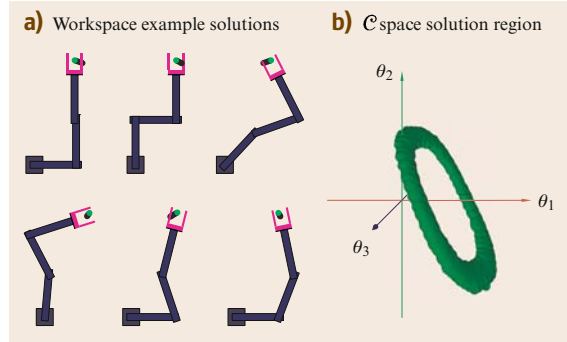


Fig. 26.13 (a) Multiple arm configurations that allow the manipulator to grasp the cylinder and (b) C-space visualization of the set of solution configurations

This means that every planning attempt with this q_{goal} will invariably fail.

Since computing a complete representation of the connectivity of \mathcal{C} is a problem of similar complexity to path planning itself, it is virtually impossible to detect a disconnection between q_{goal} and q_{init} before the planning attempt has failed.

Recall that the workspace goal for a classical manipulation planning problem is defined by a desired end-effector pose computed using inverse kinematics. Note however, that the purpose of a manipulation task may not require moving the end-effector to that particular location, but rather moving it to *any* feasible location which enables the desired manipulation task to be solved. Thus, it becomes important to devise computational methods and control schemes that are focused around the task, and allow some flexibility and freedom with regards to the end-effector pose and manipulator configuration. This is one of the motivations and advantages of

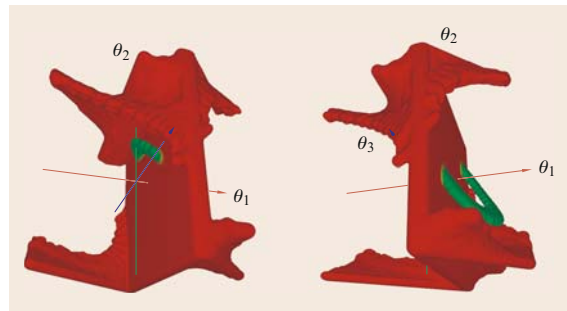


Fig. 26.14 Two views of the set of goal configurations (green) for grasping a cylindrical part in a disconnected C-space

task space or operational space control techniques (Sect. 6.2).

As an example, consider the task of grasping a cylindrical part with the planar three-DOF manipulator. Figure 26.11 shows two possible solution configurations yielding the same end-effector pose. The addition of obstacles to this workspace may cause one of the configurations to become disconnected from the zero configuration. These two configurations, however, may not be the only valid configurations that allow the manipulator to grasp the cylindrical part. Figure 26.13a shows several other configurations that allow the manipulator to grasp the part. By interpolating between these, an infinite number of different workspace poses that provide solutions to the grasping problem is generated. The inverse kinematics solutions associated with these workspace poses are contained in the continuous subset $\mathcal{C}_{\text{goal}} \subseteq \mathcal{C}$, as shown in Fig. 26.13b, which represents the actual so-

lution space to the planning problem for this reaching task.

Adding the obstacle from Fig. 26.10 to the workspace of this manipulation task produces the configuration space depicted in Fig. 26.14. Note that the \mathcal{C} -obstacle corresponding to collisions between the manipulator and the cylinder itself has been disregarded in this visualization for clarity. Approximately half of the possible solutions are disconnected from the zero configuration and a quarter are in collision, which leaves only one quarter as reachable and therefore suitable candidates for q_{goal} in a classical planning query.

In this section, we have covered the basics of manipulation planning, developed a mathematical formulation for the kinds of search spaces that arise, and discussed some of the important geometric and topological considerations. Section 26.6 gives an overview of additional topics and extensions with pointers to further reading.

26.4 Assembly Motion

Assembly or an *assembly task* defines the process of putting together manufactured parts to make a complete product, such as a machine. It is a major operation in the manufacturing process of any product. Assembly automation with robots aims to reduce cost and increase

the quality and efficiency of the operation. In environments hazardous to humans, such as in space, having robots perform assembly tasks could save human lives. Assembly has long been not only an important but also one of the most challenging applications for robotics. There are many significant research issues related to the broad scope of assembly automation, from design for assembly to tolerance analysis, assembly sequence planning, fixture design, etc. This section is only focused on the issue of robotic motion for assembly.

The concerned *assembly motion* is that of a robot manipulator holding a part and moving it to reach a certain *assembled state*, i. e., a required spatial arrangement or contact against another part. The main difficulty of assembly motion is due to the requirement for high precision or low tolerance between the parts in an assembled state. As a result, the assembly motion has to overcome uncertainty to be successful. *Compliant motion* is defined as motion constrained by the contact between the held part and another part in the environment. As it reduces uncertainty through reducing the degrees of freedom of the held part, compliant motion is desirable in assembly.

Consider the peg-in-hole insertion example introduced earlier (see Fig. 26.2). If the clearance between the peg and the hole is very small, the effect of uncertainty will most likely cause a downward insertion

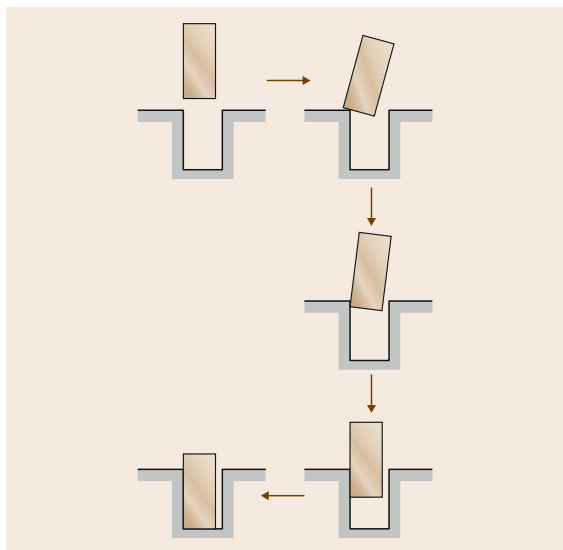


Fig. 26.15 A sequence of contact transitions for the peg-in-hole insertion task

motion of the peg to fail, i.e., the peg ends up colliding with the entrance of the hole in some way without reaching the desired assembled state. Therefore, a successful assembly motion has to move the peg out of such an unintended contact situation and lead it to reach the desired assembled state eventually. To make this transition, compliant motion is preferred. Often a sequence of contact transitions via compliant motion is necessary before the desired assembled state can be reached. Figure 26.15 shows a typical sequence of contact transitions for the peg-in-hole task.

Assembly motion strategies that incorporate compliant motion can be broadly classified into two groups: *passive compliance* and *active compliant motion*, and both groups of strategies require certain information characterizing *topological contact states* between parts.

26.4.1 Topological Contact States

When a part A contacts a part B, the configuration of A (or B) is a *contact configuration*. Often a set of contact configurations share the same high-level contact characteristics. For example, “a cup is on the table” is a high-level description shared by all the contact configurations of the cup when its bottom are on the table. Such a description is often what really matters in assembly motion as it characterizes a spatial arrangement that could be either an assembled state or just a *contact state* between a part and another part.

For contacting polyhedral objects, it is common to describe a *contact state* topologically as a set of *primitive contacts*, each of which is defined by a pair of contacting surface elements in terms of faces, edges, and vertices.

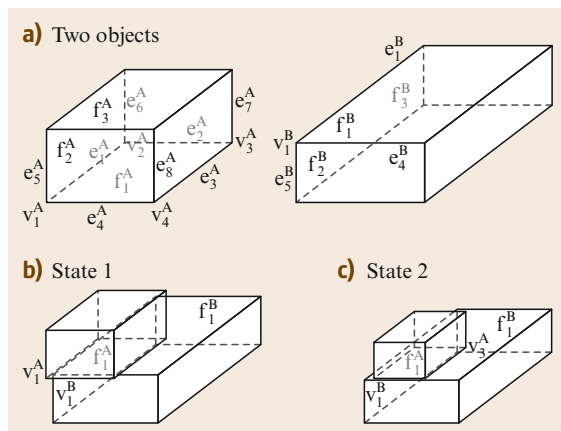


Fig. 26.16 Indistinguishable contact states due to positional uncertainty of the top part

Different contact state representations essentially differ only in how primitive contacts are defined. One common representation [26.23] defines primitive contacts as point contacts in terms of vertex–edge contacts for two-dimensional (2-D) polygons, and vertex–face and edge–edge contacts for three-dimensional (3-D) polyhedra. Another representation [26.24] defines a primitive contact as between a pair of *topological surface elements* (i.e., faces, edges, and vertices). Here a contact primitive can characterize a contact region that is either a point, a line segment, or a planar face, unlike the point-contact notion. From the viewpoint of contact identification via sensing, however, both representations can result in states that are different by definition but indistinguishable in identification due to uncertainties. Figure 26.16 shows such an example.

The notion of a *principal contact* [26.25,26] presents a high-level primitive contact that is more robust to recognition. A *principal contact* (PC) denotes a contact between a pair of topological surface elements that are not boundary elements of other contacting topological surface elements. The boundary elements of a face are the edges and vertices bounding it, and the boundary elements of an edge are the vertices bounding it. Different PCs between two objects correspond to different degrees of freedom of the objects, which often also correspond to significant differences in the contact forces and moments. As shown in Fig. 26.16, the indistinguishable states in terms of the other contact primitives are grouped as a single contact state in terms of PCs. Thus, there are also fewer contact states in terms of PCs, leading to a more concise characterization of contact states. In fact, every contact state between two convex polyhedral objects is described by a single PC.

26.4.2 Passive Compliance

Passive compliance refers to strategies that incorporate compliant motion for error correction during the assembly motion without requiring active and explicit recognition and reasoning of contact states between parts.

Remote Center Compliance

In the 1970s, a *remote center compliance* (RCC) device was developed to assist high-precision peg-in-hole insertion [26.27–29]. The RCC is a mechanical spring structure used as a tool attached to the end-effector of a robot manipulator to hold a round peg when it is inserted into a round hole. The RCC is designed to have high stiffness along the direction of insertion but high lat-

eral and angular compliances K_x and K_θ , and it projects the center of compliance near the tip of the peg (hence the name remote center compliance) to overcome small lateral and angular errors in the peg's position and orientation in response to the contact forces applied to the peg by the hole during insertion (Fig. 26.17). The large lateral and angular compliance of the RCC also helps to avoid *wedging* and *jamming*, conditions that cause the peg to stick in two-point contacts due to contact forces in direct opposition (wedging) or ill-proportioned contact forces/moments (jamming). Because the RCC is low cost and reliable for fast insertions, it has long seen successful industrial applications.

However, the RCC has limitations. It only applies to round peg-in-hole insertions, and a particular RCC works better for a peg and hole of a particular size. Attempts have been made to expand the applicability of the RCC. Some attempts try to make certain parameters of an RCC adjustable to achieve variable compliance or projection (i.e., the position of the remote center) [26.30–32]. A spatial RCC (SRCC) [26.33] was proposed to achieve square-peg-in-hole insertions. A notable difference with this extension is the combinatorial explosion of the number of possible contact states, given the range of the position and orientation uncertainty of the held part that have to be considered in the design of the device. Unlike the case of the round peg and hole, which is essentially a 2-D problem with a handful of different contact states, there are hundreds of possible different contact states between a square peg and a square hole. From each of these possible contact states, a motion leading the held part to the goal assembled state has to be realized by the device. This makes it difficult to design an RCC device that works for parts

of more general and complex shapes with more possible contact states between them. This is a major limiting factor that prevent more extensions of the RCC to assembly of different parts.

Admittance Matrix

As an alternative method to the RCC, a particular form of manipulator force control, damping control, was proposed to achieve compliant motion of the held part by the manipulator and to correct small location errors of the held part during assembly. This approach eliminates the need to build a mechanical device like an RCC to achieve error correction. Among the force control laws [26.34], damping control is a common strategy, where a commanded velocity of the held part is modified based on the sensed force caused by contact between the held part and the environment. The resulted actual velocity leads to reduction and hopefully eventual correction of small position or orientation errors of the held part. Let \mathbf{v} be a six-dimensional vector representing the actual translational and angular velocity of the held part, \mathbf{v}_0 be the six-dimensional commanded velocity, and \mathbf{f} be a six-dimensional vector representing the sensed force and moment. A linear damping control law is described as:

$$\mathbf{v} = \mathbf{v}_0 + \mathbf{A}\mathbf{f},$$

where \mathbf{A} is a 6×6 matrix, called an *admittance matrix* or *accommodation matrix*.

The effectiveness of such a damping control law depends on the existence and finding of a proper admittance matrix \mathbf{A} . There is considerable research on the design of a single \mathbf{A} that can make an assembly operation successful regardless of what contact states the held peg may encounter in the process [26.35–39]. This is aimed at cases where a single commanded velocity would be sufficient to achieve an assembly operation when there were no uncertainty or error, such as certain peg-in-hole insertion operations. One main approach to design \mathbf{A} is based on explicit kinematic and static analysis of contact conditions under all possible contact states and mating requirements, which result in a set of linear inequalities as constraints on \mathbf{A} . Learning is also used [26.38] to obtain an \mathbf{A} that minimizes the force \mathbf{f} without causing instability. Another approach [26.39] applies perturbations to the end-effector during insertion in order to obtain richer force information.

Learning Control for Assembly

Another category of approaches is to learn proper control for a particular assembly operation through stochastic or neural-network-based methods [26.40–44]. The essence

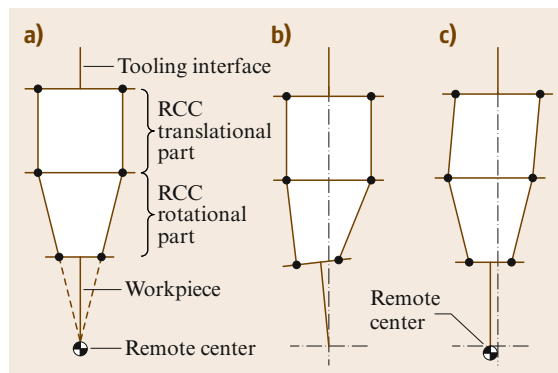


Fig. 26.17 (a) Planar representation of RCC. (b) Rotational part of RCC allowing workpiece to rotate. (c) Translational part of RCC allowing workpiece to translate

of most of these approaches is to learn to map a reaction force upon the held object caused by contact to the next commanded velocity in order to reduce errors and to achieve an assembly operation successfully. A more recent approach [26.44] maps fused sensory data of pose and vision obtained during human demonstration of assembly tasks to compliant motion signals for successful assembly.

A different approach observes assembly tasks performed by human operators through vision [26.45] or in a virtual environment [26.46, 47] and generates a motion strategy necessary for the success of the task that consists of a sequence of recognized contact state transitions and associated motion parameters.

As a sequence of commanded velocities can be generated, unlike **RCC** or strategies based on a single admittance matrix described above, can be applied to cases with large uncertainties. However, the learned controllers are task dependent.

All of the above assembly motion strategies do not require explicit recognition of contact states during their execution.

26.4.3 Active Compliant Motion

Active compliant motion is characterized by error correction based on online identification or recognition of contact states in addition to feedback of contact forces. The capability for active compliant motion allows a robot the flexibility to deal with a much broader range of assembly tasks with large uncertainties and tasks beyond assembly where compliance is required. An active compliant motion system generally requires the following components: a planner to plan compliant motion commands, an identifier to recognize contact states, state transitions, and other state information during task operation, and a controller to execute compliant motion plans based on both low-level feedback provided by sensors and high-level feedback provided by the identifier. Research on each component is described below.

Fine Motion Planning

Fine motion planning refers to planning fine-scale motions to make an assembly task successful in spite of significant uncertainties. A general approach was proposed [26.48] based on the concept of preimages in configuration space (\mathcal{C} -space) [26.23] to devise motion strategies that would not fail in the presence of uncertainties. Given a goal state of the held part defined by a region of goal configurations, a *preimage* of the goal state en-

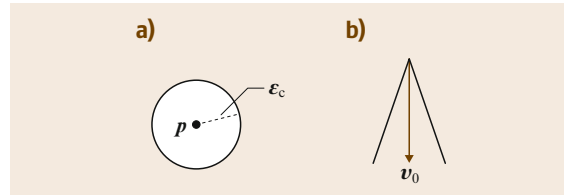


Fig. 26.18a,b Location and velocity uncertainties (in the \mathcal{C} -space). (a) The actual configuration of a part could be inside an uncertainty ball centered at an observed configuration. (b) The actual velocity of a part could be inside an uncertainty cone of a commanded velocity

codes those configurations from which a commanded velocity will guarantee that the held part reaches the goal state recognizably in spite of location and velocity uncertainties (Fig. 26.18).

Given the initial location of the held part and the goal state, the preimage approach generates a motion plan in backward chaining by finding the preimage of the goal state associated with a commanded velocity and then the preimage of the preimage, and so on, until a preimage that includes the initial configuration of the held part is found. Figure 26.19 shows an example.

The sequence of commanded velocities associated with the sequence of preimages (starting from the initial preimage that includes the initial location) forms the motion plan that guarantees the success of the task. Starting from the initial preimage, each subsequent preimage in the sequence can be viewed as a subgoal state. In this approach, compliant motions are preferred wherever possible because they typically produce larger preimages [26.48] than pure positioning motions, and subgoal states are often contact states. The approach was further extended [26.49] to include modeling uncertainties of objects.

However, the computability of the approach is a major problem. The requirement of both goal reachability

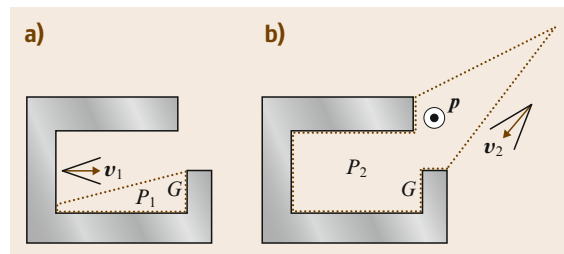


Fig. 26.19 Backward chaining of preimages: P_1 is the preimage of the goal region G , and P_2 is the preimage of P_1

and recognizability under sensing uncertainties, which are intertwined issues, complicates the computation of preimages. It has been shown [26.50] that the time complexity of generating a plan can be double exponential in nmr , where n is the number of plan steps, m is the environment complexity (of the physical space), and r is the dimension of the configuration space. By separating reachability and recognizability and restricting the recognizability power of the original preimage model, computability was improved [26.51, 52].

As an alternative, a two-phase approach is to simplify the fine motion planning into (1) global and offline nominal path planning assuming no uncertainty and (2) local and online replanning to deal with unintended contacts due to uncertainties. Different variations of the two-phase approach have been proposed [26.53–59]. The success of such an approach depends on successful online identification of contact states (see Sect. 26.4.3).

Several researchers also studied the representation and propagation of uncertainties and constraints that have to be satisfied for the success of a task [26.60–64].

Compliant Motion Planning

Compliant motion planning focuses on planning motions of objects always in contact. *Hopcroft* and *Wilfong* [26.65] proved that, if two arbitrary objects in contact can be moved to another configuration where they are also in contact, then there is always a path of contact configurations (on the boundary of configuration space obstacles, or \mathcal{C} -obstacles) connecting the first and second contact configurations. Therefore, not only is compliant motion desirable in many cases, but it is always possible given initial and goal contact configurations.

Because compliant motion occurs on the boundary of \mathcal{C} -obstacles, planning compliant motion poses special challenges not present in collision-free motion planning: it requires the exact information of contact configurations on the boundary of \mathcal{C} -obstacles. Unfortunately computing \mathcal{C} -obstacles exactly remains a formidable task to date. While there are exact descriptions of \mathcal{C} -obstacles for polygons [26.66, 67], which are three dimensional, there are only approximations of \mathcal{C} -obstacles for polyhedra [26.68, 69], which are six dimensional. If m and n indicate the complexity of two polyhedral objects in contact, the complexity of the corresponding \mathcal{C} -obstacle is $\Theta(m^6 n^6)$ [26.70].

Researchers have focused on either reducing the dimensionality and scope of the problem or getting around the problem of computing \mathcal{C} -obstacles altogether. Some researchers have studied compliant motion planning on \mathcal{C} -obstacles of reduced dimensions [26.59, 71, 72]. It is more popular to plan compliant motions based on a predetermined graph of topological contact states to avoid the problem of computing \mathcal{C} -obstacles [26.73–76]. In particular, one approach [26.74, 75] models an assembly task of polygonal parts as a discrete event system using Petri nets. However, contact states and transitions are often generated manually, which is awfully tedious for even assembly tasks of simple geometry [26.33] and is practically infeasible for complex tasks due to the huge number of different contact states.

Therefore, automatic generation of a contact state graph is desirable and even necessary. A method was first developed to enumerate all possible contact states and their connections between two convex polyhedral objects [26.77]. Recall that a contact state between two convex polyhedra can be described by a single principal contact (Sect. 26.4.1), and any principal contact between two topological surface elements of two convex polyhedra describes a geometrically valid contact state. These are nice properties and greatly simplify the problem of contact state graph generation. In general, however, to construct a contact state graph between two objects automatically requires the handling of two rather difficult issues:

1. how to generate *valid* contact states, i. e., how to tell if a set of principal contacts corresponds to a geometrically valid contact state, given the geometries of objects, and
2. how to link one valid contact state to another in the graph, i. e., how to find the neighboring (or adjacency) relations among the regions of contact

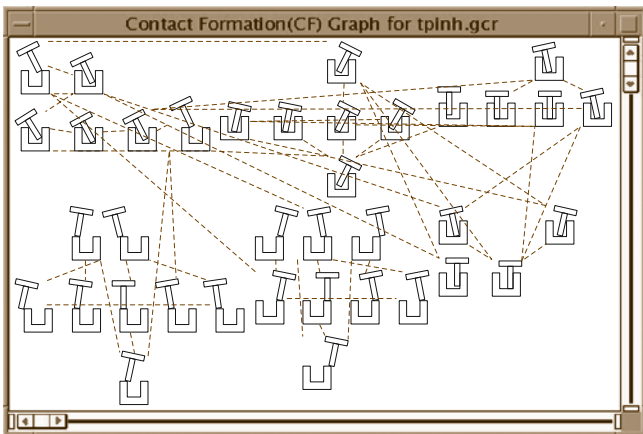


Fig. 26.20 A contact state graph between two planar parts

configurations belonging to different contact states in the contact configuration space.

A general and efficient divide-and-merge approach was introduced [26.26] for automatically generating a contact state graph between two arbitrary polyhedra. Each node in the graph denotes a contact state, described by a topological *contact formation* (CF) [26.25] as a set of principal contacts and a configuration satisfying the CF. Each edge connects the nodes of two neighboring contact states. Figure 26.20 shows an example contact state graph between two planar parts.

The approach handled the above two issues simultaneously by directly exploiting both topological and geometrical knowledge of contacts in the *physical space* of objects and by dividing the problem into simpler subproblems of generating and merging special subgraphs. Specifically, the approach takes advantage of the fact that a contact state graph can be divided into special subgraphs called the *goal-contact relaxation* (GCR) graphs, where each GCR graph is defined by a locally most constrained valid contact state, called the seed, and its less-constrained neighboring valid contact states, which is easier to generate because of several properties. The main properties include:

- given a valid contact state, CS_i , all of its less constrained neighboring contact states can be hypothesized topologically from the principal contacts in CS_i ,
- a hypothesized less constrained neighboring contact state, CS_j , is valid, if and only if there exists a compliant motion to relax certain constraints of CS_i to obtain CS_j which does not result in any other contact state, called neighboring relaxation,
- neighboring relaxation can often be achieved by instantaneous compliant motion.

With this approach, a contact state graph of several hundreds or thousands of nodes and links can be generated in a few seconds.

With a contact state graph, the problem of compliant motion planning can be decomposed into two simpler subproblems at two levels: (1) high level: graph search for state transitions from one node to another in a contact state graph, and (2) low level: contact motion planning within the set of contact configurations constrained by the *same* contact state (and the same contact formation), called *CF-compliant motion planning*. A general contact motion plan crossing several contact states can be considered as consisting of segments of

CF-compliant motions in different contact states. One approach plans [26.78] CF-compliant motions based on random sampling of CF-compliant configurations and extending the probabilistic roadmap motion planning technique [26.79].

Contact State Identification

Successful execution of a fine-motion or compliant motion plan depends on correct online identification of contact states during execution. Contact state identification uses both model information of contact states (including topological, geometrical and physical information) and sensory information of location and force/torque of the end-effector. The presence of sensing uncertainties make the identification problem nontrivial. Approaches for contact state identification are different in ways of dealing with uncertainties.

One class of approaches obtain the mapping between sensory data (in the presence of sensing uncertainties) and corresponding contact states through learning. Models for learning include hidden Markov models [26.80, 81], thresholds [26.82, 83], neural network structures, and fuzzy classifiers [26.84–88]. Training data are obtained either in an unsupervised way or by human task demonstration. Such approaches are task dependent: a new task or environment requires new training.

Another class of approaches are based on analytical models for contact states. A common strategy is to predetermine the set of configurations, constraints on configurations, or the set of forces/torques or force/torque constraints that are possible for each contact state and match such information against the sensed data to identify contact states online. Some approaches do not consider the effect of uncertainties [26.89, 90], while others consider uncertainties modeled either by uncertainty bounds or by probability distributions [26.24, 76, 91–95]. An alternative strategy is based on checking the distance between contacting elements in Cartesian space: either by growing the objects with pose uncertainty and intersecting the obtained regions [26.96], or by combining the distance between the objects with several other factors such as the instantaneous approach direction [26.97].

In terms of how sensory data are used, some identification schemes are rather static because they do not consider the prior motion and state identifications before the current contact state is reached, while others use prior history or even use new motion or *active sensing* to help current identification. The latter include approaches that perform simultaneous contact state identification and

parameter estimation. In such approaches, an analytical contact state model is expressed as a function of uncertain parameters, such as the pose or dimension of an object in contact. At each contact state, some of the uncertain parameters can be observed or estimated with reduced uncertainty. For example, if the held object is at a face–face contact with another object, parameters describing the normals of the contacting faces can be estimated with the help of force/torque sensing. During a task execution (i. e., the execution of a motion plan), the uncertain parameters are estimated along with the identification of contact states with increasing accuracy, which in turn makes the subsequent contact state identifications more accurate. This also improves force control and contact state transition monitoring. The increased performance, however, comes at a higher computational cost.

Simultaneous contact state identification and parameter estimation is most often done by ruling out contact state models that are inconsistent with sensed data or parameter estimation results [26.98–104]. One notable exception [26.105] performs a truly simultaneous estimation by describing the system as a hybrid joint probability distribution of contact states and parameters.

Active sensing is about deliberate use of applied force/torque or motion to better aid contact state identification and parameter estimation. Earlier research focused on simple force strategy [26.106] or movability tests [26.24, 97]. More recently, methods were introduced [26.107] to design sequences of contact state transitions and compliant motion strategies that were optimized for active sensing to determine all uncertain geometric parameters while achieving certain goal contact state.

From a different perspective, an approach was introduced to analyze contact state distinguishability and unknown/uncertain parameter identifiability [26.108]

during the design phase of a contact state identifier or parameter estimator.

Execution of Compliant Motion Plans

While compliant or force control is a much researched subject (see Chap. 8), how to execute a compliant motion plan automatically has received much less attention until recently.

A general compliant motion plan (as the output of a compliant motion or fine motion planner) usually consists of a sequence of contact states, and within each contact state, a path of contact configurations compliant to the contact state and leads to the next contact state in the sequence. Such topological and geometrical information alone is not sufficient for a compliant controller to execute the plan, especially when there are many different and complex contact states.

A hybrid position/force controller requires a specification that separates dimensions of force control from those of position/velocity control. Such specifications not only have to vary for different contact states but may have to vary from one contact configuration to another within the same contact state, i. e., a control specification is a function of time or configuration along a compliant trajectory in general. Moreover, it is far from trivial to make control specifications for complex contact states involving multiple principal contacts [26.109].

Recently an approach was introduced [26.110] to convert automatically a compliant path of contact configurations across a sequence of contact states into wrench, twist, and position control signals $w(t)$, $t(t)$, and $p(t)$ for a hybrid controller to execute the plan. The approach was experimentally verified successfully.

However, there is not yet a full integration of planning, online contact state identification, online replanning (to deal with contact states that are off the preplanned path due to uncertainty), and compliant motion control.

26.5 Unifying Feedback Control and Planning

In previous sections we discussed task-level control methods and planning algorithms. Both task-level control and planning methods are able to address specific constraints imposed on robot motion in the context of manipulation tasks. However, as illustrated in Fig. 26.3, neither of these categories of methods is able to address all constraints completely. Control remains susceptible to local minima and thus cannot guarantee that a particular motion will lead to the desired result. Planning

methods, on the other hand, overcome the problem of local minima by projecting possible motions into the future to predict if a sequence of motion will lead to success. Given certain assumptions, the resulting plan can be guaranteed to succeed. However, the computations associated with this process are generally too computationally complex to satisfy the feedback requirements of manipulation tasks. Unfortunately, this means that control and motion planning by themselves are unable

to address the problem of moving robots for general manipulations tasks.

In this final section of this chapter, we will review efforts to combine the advantages of control methods and planning methods into a single approach to determine the motion of a robot. These efforts aim to create methods that avoid the susceptibility to local minima while satisfying the feedback requirements. Early attempts of integrating planning and control occurred in the early 1990s. Not all of these efforts are specifically directed at manipulation but many contain insights relevant to the topic. In this section, we will review these efforts and also discuss some of the more recent research efforts aimed at unifying motion planning and feedback control.

Motion planning (Chap. 5) and motion control (Chap. 6) have traditionally been regarded as two distinct areas of research. However, these areas have many features in common. Both areas are concerned with the motion of robotic mechanisms, and more importantly planning and control methods both determine a representation that maps robot state to robot motion. In the case of feedback control, this representation is a potential function defined for a given region of the state space. The gradient of the potential function at a specific state encodes the motion command. In contrast, motion planning determines motion plans. These motion plans also encode motions for a set of states.

Motion planning and feedback control also differ in certain aspects. A motion planner generally makes stronger assumptions than a controller about the environment, the ability to assess its state, and the changes that can occur in the environment. Motion planning also requires the ability to project the robot's state into the future, given its current state and a particular action. By using this ability together with global information about the environment, motion planners can determine motions that are not susceptible to local minima. This positive characteristic of motion planners, however, results in significantly increased computational cost (see Chap. 5). The large discrepancy in computational requirements for planning and control and the resulting divergence of computational techniques may explain the current separation of the two fields. Researchers are beginning to attempt to reverse this separation by devising a unified theory of planning and control.

26.5.1 Feedback Motion Planning

Feedback motion planning combines planning and feedback into a single motion strategy. A planner considers

global information to compute a feedback motion plan free of local minima. Such a feedback motion plan can be interpreted as a potential function or vector field whose gradient will lead the robot to the goal state from any reachable part of the state space [26.18]. Local minima-free potential functions are also called navigation functions [26.111, 112]. Given the current state of the robot and the global navigation function, feedback control is used to determine the robot's motion. The consideration of feedback about the robot's state in the context of a global navigation function reduces the susceptibility to sensing and actuation uncertainty.

Feedback motion planning, in principle, can address the entire spectrum of motion constraints and their feedback requirements (see Fig. 26.3). Given a global navigation function that considers all motion constraints, the feedback requirements can easily be satisfied, since the feedback motion plan already specifies the desired motion command for the entire state space. Obviously, the major challenge in feedback motion planning is the computation of such a navigation function or feedback motion plan. The problem becomes particularly difficult in the context of a manipulation task, since the state space (or configuration space) changes each time the robot grasps or releases an object in the environment (Sect. 26.3). This change makes it necessary to recompute the feedback motion plan. Frequent recomputation is also necessary in dynamic environments, where the motion of obstacles can repeatedly invalidate a previously computed feedback motion plan.

In the remainder of this section, we will review a variety of methods for computing navigation functions. In general, the problem of efficiently computing feedback motion plans for manipulation tasks remains unsolved. We therefore also present methods that do not explicitly consider manipulation. These methods can be divided into three categories: (1) exact methods, (2) approximate methods based on dynamic programming, and (3) approximate methods based on composing and sequencing simpler potential functions.

The earliest exact methods for the computation of navigation functions are applicable to simple environments with obstacles of specific shapes [26.111–113]. Approximate methods based on discretized spaces (grids) overcome this limitation but possess an exponential computational complexity in the number of dimensions of the state space. These approximate navigation functions are called numerical navigation functions [26.114]. These navigation functions are often used for applications in mobile robotics. Due to the low-dimensional configuration space associated with

mobile robots, they can generally solve motion planning problems robustly and efficiently.

Some physical processes, such as heat transfer or fluid flow, can be described by a specific type of differential equation, called harmonic functions. These functions possess properties that make them suitable as navigation functions [26.115–119]. Navigation functions based on harmonic functions are most commonly computed in an approximate, iterative fashion. The requirement for iterative computation increases the computational cost relative to the simpler numerical navigation functions [26.114].

More recent methods for the computation of numerical navigation functions consider differential motion constraints at a significantly reduced computational cost [26.120]. These methods rely on classical numerical dynamic programming techniques and numerical optimal control. However, in spite of their reduced computational complexity, these methods remain too computationally costly to be applied to manipulation tasks with many degrees of freedom in dynamic environments.

Navigation functions can also be computed by composing local potential functions based on global information. This is illustrated in Fig. 26.21. The goal is to compute a navigation function for the entire configuration space C . This is accomplished by sequencing overlapping funnels. Each of the funnels represents a simple, local potential function. By following the gradient of this potential function, motion commands can be determined for a subset of the configuration space. If the funnels are sequenced correctly, the composition of local funnels can yield a global feedback plan. This feedback plan can be viewed as a hybrid system [26.121] in which the sequencing of funnels represents a discrete transition structure, whereas the individual controllers operate in a continuous domain. The composition of funnels is considered planning. The consideration of global information during the planning permits to determine a funnel composition that avoids local minima.

One of the earliest methods based on the global composition of local funnels was proposed by Choi et al. [26.122]. In this method, the state space of the robot is decomposed into convex regions. The connectivity of these regions is analyzed to determine global information about the state space. This information can be used to combine simple, local potential functions, one for each convex regions of state space, into a local minima-free potential function. More rigorous approaches based on this idea have been developed. These approaches can consider the dynamics of the robot and nonholonomic

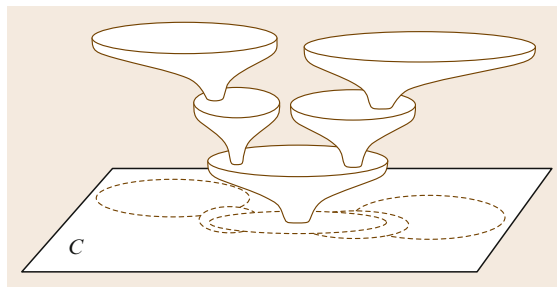


Fig. 26.21 Composing funnels into a navigation function

motion constraints [26.123–125]. These methods have only been applied to low-dimensional state spaces and cannot easily be applied to manipulation tasks.

The random neighborhood graph [26.126] is a sampling-based method of computing a decomposition of the overall state space. As before, a navigation function can be computed by analyzing the global connectivity of the decomposition and imposing adequate local potential functions for each subdivision. A specialized method following the same principle for planar robots in polygonal environments has also been proposed [26.127]. The idea of composing local potential functions has also been applied successfully to a complex robot control task [26.128]. Finally, in [26.129] a general and efficient method of computing a smooth feedback plan over a cylindrical algebraic decomposition of configuration space has been proposed.

The computation of a navigation function over the entire configuration space quickly becomes intractable as the dimensionality of the space increases. To overcome this challenge, in particular in the context of autonomous mobile manipulation, workspace heuristics have been employed to determine a navigation function efficiently. This navigation function does not cover the entire configuration space but only those regions heuristically determined to be relevant to the motion problem [26.130]. This method of generating feedback plans is able to satisfy the various motion constraints depicted in Fig. 26.3 and their respective feedback requirements.

26.5.2 Augmenting Global Plans with Feedback

The manipulation planning techniques described in Sect. 26.3 are not susceptible to local minima, as they consider global state space information. Due to the computational complexity associated with the consideration of global information, these planning techniques are

not able to satisfy the feedback requirements of manipulation tasks. However, the required frequency of feedback about global motion are relatively low: the global connectivity of the configuration space changes relatively infrequently (Fig. 26.3). It would therefore be possible to consider feedback for global motion at the slow rates the planner can accommodate, while considering feedback for other motion constraints at higher frequencies. To achieve this, global motion plans have to be augmented with reactive components that incrementally modify the global plan in response to feedback from the environment. As long as the global connectivity information captured by the plan remains valid, the incremental modifications can ensure that all other motion constraints, ranging from task requirements to reactive obstacles avoidance, are satisfied.

The elastic-band framework [26.131] augments global plans with reactive obstacle avoidance. A global configuration space path, determined by a planner, is covered with local potential functions, each of which is derived from the local distribution of obstacles around the path. These local potentials cause the path to deform so as to maintain a minimum distance from obstacles. Visually, the path behaves as an elastic band that is deformed by the motion of obstacles. The local potential functions, together with the global path, can be viewed as a navigation function for a local region of the configuration space. Integrated with a global planner and replanner, the elastic-band framework permits real-time obstacle avoidance that is not susceptible to local minima. However, the feedback frequency for global motion remains limited by the global motion planner. Specific task constraints have

not been integrated into the elastic-band framework; consequently, its application to manipulation tasks is limited.

In its original formulation, the elastic-band framework assumed that all degrees of freedom of the robot are holonomic. An extended formulation augments motion paths for nonholonomic platforms with reactive components [26.132].

The elastic-strip framework [26.133] also augments global motion plans with reactive obstacle avoidance. In addition to reactive obstacle avoidance, however, the elastic-strip framework can accommodate task constraints. Similarly to an elastic band, an elastic strip covers a global path with local potential functions. In contrast to the elastic-band framework, these potential functions are based on task-level controllers (Sect. 26.2) and therefore allow the task-consistent modification of the global path. The elastic-strip framework is therefore well suited for the execution of manipulation plans in dynamic environments. An elastic strip will be incrementally modified to represent a constraint-consistent trajectory, as long as the global information captured by the underlying plan remains valid. The elastic-strip framework has been applied to a variety of manipulation tasks on a mobile manipulation platform.

Extending the elastic-band and elastic-strips frameworks, the elastic-roadmap framework combines reactive task-level control with efficient global motion planning [26.130]. The elastic roadmap represents a hybrid system of task-level controllers that are composed into a navigation function, thereby satisfying the motion constraints depicted in Fig. 26.3 and their respective feedback requirements.

26.6 Conclusions and Further Reading

In this chapter, an overview of motion generation and control strategies in the context of robotic manipulation tasks has been provided. Issues related to modeling the interfaces between the robot and the environment at the different time scales of motion and incorporating sensing and feedback were considered. Manipulation planning was introduced as an extension to the basic motion planning problem, which can be modeled as a hybrid system of continuous configuration spaces arising from the act of grasping and moving parts in the environment. The important example of assembly motion has been discussed through the analysis of contact states and compliant motion control. The operational space framework, as described in Sect. 26.2, permits the position and

force control of operational points on a serial-chain manipulator. A number of extensions have been proposed in the literature, extending the framework to situations with multiple concurrent contact points, cooperative manipulation scenarios, and branching kinematic chains. For manipulation planning, extensions involving grasp and regrasp planning, multiple robots, multiple parts, and movable obstacles have been considered. In this section we will briefly discuss these extensions and refer to the appropriate literature.

General Contact Models

The hybrid force/motion control described by *Raibert* and *Craig* [26.134] has been shown to have shortcom-

ings [26.135]. A general contact model for dynamically decoupled force/motion control in the context of operational space control that overcomes these problems has been presented in [26.135]. This general contact model has been extended into a compliant motion control framework to enable force control for multiple contact points [26.136] in nonrigid environments, and to enable force control for multiple contact points on different links of a kinematic chain [26.137]. The operational space framework for task-level control has been applied for the real-time simulation of complex dynamic environments [26.138]. Contact points between objects in the environment are modeled as operation points, resulting in a mathematically elegant framework for resolving impulses and contact constraints for moving bodies.

Cooperative Manipulation Control

If multiple task-level controlled robots collaborate to manipulate an object, they form a closed kinematic chain that is connected through the object. The dynamics of the entire system can be described using the notion of an augmented object [26.139, 140], in which the dynamics of the manipulators and the object are combined to form a model of the overall system. The internal forces that occur at the grasp points during motion can be modeled using the virtual linkage framework [26.141]. A number of alternative strategies for the cooperative manipulation of multiple robots outside of the operational space framework have been proposed [26.142–148].

Control of Branching Mechanisms

So far, we have implicitly assumed that the task-controlled robot consists of a single kinematic chain. This assumption does not hold in the case of kinematically more complex mechanisms, such as humanoid robots (Chap. 56). These robots can consist of multiple branching kinematic chains. If we consider the torso of a humanoid robot to be the robot's base, for example, then the legs, arms, and the head represent five kinematic chains attached to this base. We call such a mechanism a branching mechanism if it does not contain any closed kinematic loops. Tasks performed by a branching mechanism may require the specification of operational points on any one of those branches; for example, while the legs perform locomotion the hands achieve a manipulation task and the head is oriented to maintain visibility of the manipulated object.

The operational space framework has been extended to task-level control of branching mechanisms [26.149]. This extension combined operational points and associated Jacobians and computes an operational space

inertia matrix that combines all operational points. This matrix can be computed efficiently with an algorithm that in practice is linear in the number of operational points [26.150, 151].

Nonholonomic Mobile Manipulation

All previously discussed work on task-level control assumes that the degrees of freedom are holonomic. For manipulator arms this is generally a valid assumption. The most common type of mobility platform, however, is based on either differential drives, synchrodrives, or Ackerman steerings (point to a section), all of which are subject to nonholonomic constraints. The operational space framework has been extended to mobile manipulation platforms that combine a nonholonomic mobility platform with a holonomic manipulator arm [26.152–154], enabling the task-level control of a large class of mobile manipulation platforms.

Learning Models with Uncertainty

The efficacy of operational space control depends on the accuracy of the dynamic model of the robot. In particular when multiple behaviors are executed using null-space projections, modeling errors can have significant effects. To overcome the reliance on accurate dynamic models, reinforcement learning can be used to learn operational space controllers [26.155].

Grasping and Regrasp Planning

Deciding the intermediate stable configurations for a part necessary to complete a manipulation task and selecting proper grasps are both difficult problems in themselves that operate over a continuum of possible solutions. *Simeon* et al. developed a manipulation planning framework that considers continuous grasps and placements [26.156]. However, selecting from among all possible grasp configurations and deciding when to regrasp is still an open research problem. For an overview of various grasp quality metrics, see *Miller* and *Allen* [26.157, 158] and Chap. 28.

Multiple Parts

The manipulation planning framework nicely generalizes to multiple parts, $\mathcal{P}_1, \dots, \mathcal{P}_k$. Each part has its own \mathcal{C} -space, and \mathcal{C} is formed by taking the Cartesian product of all the part \mathcal{C} -spaces with the manipulator \mathcal{C} -space. The set \mathcal{C}_{adm} is defined in a similar way, but now part–part collisions also have to be removed, in addition to part–manipulator, manipulator–obstacle, and part–obstacle collisions. The definition of $\mathcal{C}_{\text{stable}}$ re-

quires that all parts be in stable configurations; the parts may even be allowed to stack on top of each other. The definition of $\mathcal{C}_{\text{grasp}}$ requires that one part is grasped and all other parts are stable. There are still two modes, depending on whether the manipulator is grasping a part. Once again, transitions occur only when the robot is in $\mathcal{C}_{\text{trans}} = \mathcal{C}_{\text{stable}} \cap \mathcal{C}_{\text{grasp}}$.

Planning for Multiple Robots

Generalizing to k robots would lead to $2k$ modes, in which each mode indicates whether each robot is grasping a part. Multiple robots may even be allowed to grasp the same part, which leads to the interesting problem of planning for closed kinematic chains and cooperative motion (see Chap. 29). Koga addressed multi-arm manipulation planning where the same object must be grasped and moved by several manipulator arms [26.21]. Another generalization could allow a single robot to grasp more than one part simultaneously.

Planning for Closed Kinematic Chains

The subspace of \mathcal{C} that results from maintaining kinematic closure arises when multiple robots grasp the same part, or even when multiple fingers of the same hand grasp a single part (see Chap. 15). Planning in this context requires that paths remain on a lower-dimensional variety for which a parameterization is not available. Planning the motion of parallel mechanisms or other systems with loops typically requires maintaining multiple closure constraints simultaneously (Chap. 12).

Planning with Movable Obstacles

In some cases, the robot may be allowed to reposition obstacles in the environment rather than simply avoid them. Wilfong first addressed motion planning in the presence of movable obstacles [26.159]. Wil-

fong proved that the problem is PSPACE-hard even in two-dimensional environments where the final positions of all movable objects are specified. Erdmann and Lozano-Perez considered coordinated planning for multiple moving objects [26.160]. Alami presented a general algorithm for the case of one robot and one movable object and formulated the space of grasping configurations into a finite number of cells [26.17]. Chen and Hwang developed a planner for a circular robot that is allowed to push objects aside as it moves [26.161]. Stilman and Kuffner considered movable obstacles in the context of navigation planning [26.162], and manipulation planning [26.163]. Nieuwenhuisen et al. also developed a general framework for planning with movable obstacles [26.164].

Nonprehensile Manipulation

Lynch and Mason explored scenarios in which grasping operations are replaced by pushing operations [26.165]. The space of stable pushing directions imposes non-holonomic constraints on the motion of the robot, which opens up issues related to controllability. Related issues also arise in the context of part-feeders and manipulation for manufacturing and assembly (Chap. 42).

Assembly Motion Extensions

The work in assembly motion described in this chapter has focused on rigid-part assembly, and the parts considered are mostly polyhedral or of simple non-polyhedral shapes, such as round pegs and holes. More recent work on contact state analysis for curved objects can be found in [26.166]. An emerging field in assembly is micro/nanoassembly (Chap. 18). Flexible or deformable part assembly has also begun to attract attention [26.167–169]. Virtual assembly [26.170] for simulation and prototyping is another interesting area of study.

References

- 26.1 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)
- 26.2 H. Inoue: Force feedback in precise assembly tasks, Tech. Rep. **308** (Artificial Intelligence Laboratory, MIT, Cambridge 1974)
- 26.3 T. Lozano-Pérez, M. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robots, Int. J. Robot. Res. **31**(1), 3–24 (1984)
- 26.4 O. Khatib: A unified approach to motion and force control of robot manipulators: the operational space formulation, Int. J. Robot. Autom. **3**(1), 43–53 (1987)
- 26.5 O. Khatib, K. Yokoi, O. Brock, K.-S. Chang, A. Casal: Robots in human environments, Arch. Contr. Sci. **11**(3/4), 123–138 (2001)
- 26.6 G. Strang: *Linear Algebra and Its Applications* (Brooks Cole, New York 1988)
- 26.7 H. Seraji: An on-line approach to coordinated mobility and manipulation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Vol. 1 (Atlanta 1993) pp. 28–35
- 26.8 J.M. Cameron, D.C. MacKenzie, K.R. Ward, R.C. Arkin, W.J. Book: Reactive control for mobile manipulation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Vol. 3 (Atlanta 1993) pp. 228–235

- 26.9 M. Egerstedt, X. Hu: Coordinated trajectory following for mobile manipulation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (San Francisco 2000)
- 26.10 P. Ögren, M. Egerstedt, X. Hu: Reactive mobile manipulation using dynamic trajectory tracking, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (San Francisco 2000) pp. 3473–3478
- 26.11 J. Tan, N. Xi, Y. Wang: Integrated task planning and control for mobile manipulators, Int. J. Robot. Res. **22**(5), 337–354 (2003)
- 26.12 Y. Yamamoto, X. Yun: Unified analysis on mobility and manipulability of mobile manipulators, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Detroit 1999) pp. 1200–1206
- 26.13 L. Sentis, O. Khatib: Control of free-floating humanoid robots through task prioritization, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Barcelona 2005)
- 26.14 L. Sentis, O. Khatib: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives, Int. J. Human Robot. **2**(4), 505–518 (2005)
- 26.15 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, Int. J. Robot. Res. **5**(1), 90–98 (1986)
- 26.16 M. Huber, R.A. Grupen: A feedback control structure for on-line learning tasks, Robot. Auton. Syst. **22**(3–4), 303–315 (1997)
- 26.17 R. Alami, J.P. Laumond, T. Siméon: Two manipulation planning algorithms, Workshop Algorithm. Found. Robot. (1994)
- 26.18 S.M. LaValle: *Planning Algorithms* (Cambridge Univ. Press, Cambridge 2006), (also available at <http://msl.cs.uiuc.edu/planning/>)
- 26.19 R. Grossman, A. Nerode, A. Ravn, H. Rischel (Eds.): *Hybrid Systems* (Springer, Berlin, Heidelberg 1993)
- 26.20 K.J. Gupta Ahuactzin, E. Mazer: Manipulation planning for redundant robots: a practical approach, Int. J. Robot. Res. **17**(7), 731–747 (1998)
- 26.21 Y. Koga: On Computing Multi-Arm Manipulation Trajectories. Ph.D. Thesis (Stanford University, Stanford 1995)
- 26.22 D. Bertram, J.J. Kuffner, T. Asfour, R. Dillman: A unified approach to inverse kinematics and path planning for redundant manipulators, Proc. IEEE Int. Conf. Robot. Autom. (ICRA'06) (2006) pp. 1874–1879
- 26.23 T. Lozano-Pérez: Spatial planning: a configuration space approach, IEEE Trans. Comput. **C-32**(2), 108–120 (1983)
- 26.24 R. Desai: On Fine Motion in Mechanical Assembly in Presence of Uncertainty. Ph.D. Thesis (Department of Mechanical Engineering, University of Michigan, 1989)
- 26.25 J. Xiao: Automatic determination of topological contacts in the presence of sensing uncertainties, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Atlanta 1993) pp. 65–70
- 26.26 J. Xiao, X. Ji: On automatic generation of high-level contact state space, Int. J. Robot. Res. **20**(7), 584–606 (2001), (and its first multi-media extension issue eIJRR)
- 26.27 S.N. Simunovic: Force information in assembly processes, Proc. 5th Int. Symp. Ind. Robots (1975) pp. 415–431
- 26.28 S.H. Drake: Using Compliance in Lieu of Sensory Feedback for Automatic Assembly. Ph.D. Thesis (Department of Mechanical Engineering, Massachusetts Institute of Technology 1989)
- 26.29 D.E. Whitney: Quasi-static assembly of compliantly supported rigid parts, ASME J. Dyn. Syst. Meas. Contr. **104**, 65–77 (1982)
- 26.30 R.L. Hollis: A six-degree-of-freedom magnetically levitated variable compliance fine-motion wrist: design, modeling, and control, IEEE Trans. Robot. Autom. **7**(3), 320–332 (1991)
- 26.31 S. Joo, F. Miyazaki: Development of variable RCC and ITS application, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS), Vol. 3 (Victoria 1998) pp. 1326–1332
- 26.32 H. Kazerooni: Direct-drive active compliant end effector (active RCC), IEEE J. Robot. Autom. **4**(3), 324–333 (1988)
- 26.33 R.H. Sturges, S. Laowattana: Fine motion planning through constraint network analysis, Proc. Int. Symp. Assem. Task Plan. (1995), 160–170
- 26.34 D.E. Whitney: Historic perspective and state of the art in robot force control, Int. J. Robot. Res. **6**(1), 3–14 (1987)
- 26.35 M. Peshkin: Programmed compliance for error corrective assembly, IEEE Trans. Robot. Autom. **6**(4), 473–482 (1990)
- 26.36 J.M. Schimmels, M.A. Peshkin: Admittance matrix design for force-guided assembly, IEEE Trans. Robot. Autom. **8**(2), 213–227 (1992)
- 26.37 J.M. Schimmels: A linear space of admittance control laws that guarantees force assembly with friction, IEEE Trans. Robot. Autom. **13**(5), 656–667 (1997)
- 26.38 S. Hirai, T. Inatsugi, K. Iwata: Learning of admittance matrix elements for manipulative operations, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (1996) pp. 763–768
- 26.39 S. Lee, H. Asada: A perturbation/correlation method for force guided robot assembly, IEEE Trans. Robot. Autom. **15**(4), 764–773 (1999)
- 26.40 H. Asada: Representation and learning of nonlinear compliance using neural nets, IEEE Trans. Robot. Autom. **9**(6), 863–867 (1993)
- 26.41 J. Simons, H. Van Brussel, J. De Schutter, J. Verhaert: A self-learning automaton with variable resolution for high precision assembly by industrial robots, IEEE Trans. Autom. Contr. **27**(5), 1109–1113 (1982)

- 26.42 V. Gullapalli, J.A. Franklin, H. Benbrahim: Acquiring robot skills via reinforcement learning, IEEE Contr. Syst. **14**(1), 13–24 (1994)
- 26.43 Q. Wang, J. De Schutter, W. Witvrouw, S. Graves: Derivation of compliant motion programs based on human demonstration, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1996) pp. 2616–2621
- 26.44 R. Cortesao, R. Koeppe, U. Nunes, G. Hirzinger: Data fusion for robotic assembly tasks based on human skills, IEEE Trans. Robot. Autom. **20**(6), 941–952 (2004)
- 26.45 K. Ikeuchi, T. Suehiro: Toward an assembly plan from observation. Part I: task recognition with polyhedral objects, IEEE Trans. Robot. Autom. **10**(3), 368–385 (1994)
- 26.46 H. Onda, H. Hirokawa, F. Tomita, T. Suehiro, K. Takase: Assembly motion teaching system using position/forcesimulator-generating control program, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (1997) pp. 938–945
- 26.47 H. Onda, T. Suehiro, K. Kitagaki: Teaching by demonstration of assembly motion in VR – non-deterministic search-type motion in the teaching stage, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (2002) pp. 3066–3072
- 26.48 T. Lozano-Pérez, M.T. Mason, R.H. Taylor: Automatic synthesis of fine-motion strategies for robot, Int. J. Robot. Res. **31**(1), 3–24 (1984)
- 26.49 B.R. Donald: *Error Detection and Recovery in Robotics* (Springer, Berlin, Heidelberg 1989)
- 26.50 J. Canny: On computability of fine motion plans, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1989) pp. 177–182
- 26.51 M. Erdmann: Using backprojections for fine motion planning with uncertainty, Int. J. Robot. Res. **5**(1), 19–45 (1986)
- 26.52 J.C. Latombe: *Robot Motion Planning* (Kluwer Academic, Dordrecht 1991)
- 26.53 B. Dufay, J.C. Latombe: An approach to automatic programming based on inductive learning, Int. J. Robot. Res. **3**(4), 3–20 (1984)
- 26.54 H. Asada, S. Hirai: Towards a symbolic-level force feedback: recognition of assembly process states, Proc. Int. Symp. Robot. Res. (1989) pp. 290–295
- 26.55 J. Xiao, R. Volz: On replanning for assembly tasks using robots in the presence of uncertainties, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1989) pp. 638–645
- 26.56 J. Xiao: Replanning with compliant rotations in the presence of uncertainties, Proc. Int. Symp. Intell. Contr. (Glasgow 1992) pp. 102–107
- 26.57 G. Dakin, R. Popplestone: Simplified fine-motion planning in generalized contact space, Proc. Int. Symp. Intell. Contr. (1992) pp. 281–287
- 26.58 G. Dakin, R. Popplestone: Contact space analysis for narrow-clearance assemblies, Proc. Int. Symp. Intell. Contr. (1993) pp. 542–547
- 26.59 J. Rosell, L. Basañez, R. Suárez: Compliant-motion planning and execution for robotic assembly, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1999) pp. 2774–2779
- 26.60 R. Taylor: The Synthesis of Manipulator Control Programs from Task-Level Specifications. Ph.D. Thesis (Stanford University, Stanford 1976)
- 26.61 R.A. Brooks: Symbolic error analysis and robot planning, Int. J. Robot. Res. **1**(4), 29–68 (1982)
- 26.62 R.A. Smith, P. Cheeseman: On the representation and estimation of spatial uncertainty, Int. J. Robot. Res. **5**(4), 56–68 (1986)
- 26.63 S.-F. Su, C. Lee: Manipulation and propagation of uncertainty and verification of applicability of actions in assembly tasks, IEEE Trans. Syst. Man Cybern. **22**(6), 1376–1389 (1992)
- 26.64 S.-F. Su, C. Lee, W. Hsu: Automatic generation of goal regions for assembly tasks in the presence of uncertainty, IEEE Trans. Robot. Autom. **12**(2), 313–323 (1996)
- 26.65 J. Hopcroft, G. Wilfong: Motion of objects in contact, Int. J. Robot. Res. **4**(4), 32–46 (1986)
- 26.66 F. Avnaim, J.D. Boissonnat, B. Faverjon: A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1988) pp. 1656–1661
- 26.67 R. Brost: Computing metric and topological properties of c-space obstacles, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1989) pp. 170–176
- 26.68 B. Donald: A search algorithm for motion planning with six degrees of freedom, Artif. Intell. **31**(3), 295–353 (1987)
- 26.69 L. Joskowicz, R.H. Taylor: Interference-free insertion of a solid body into a cavity: an algorithm and a medical application, Int. J. Robot. Res. **15**(3), 211–229 (1996)
- 26.70 H. Hirukawa: On motion planning of polyhedra in contact, Workshop Algorithm. Fund. Robot. (Toulouse 1996) pp. 381–391
- 26.71 S.J. Buckley: Planning compliant motion strategies, Proc. Int. Symp. Intell. Contr. (1988) pp. 338–343
- 26.72 E. Sacks: Path planning for planar articulated robots using configuration spaces and compliant motion, IEEE Trans. Robot. Autom. **19**(3), 381–390 (2003)
- 26.73 C. Laugier: Planning fine motion strategies by reasoning in the contact space, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1989) pp. 653–659
- 26.74 B.J. McCarragher, H. Asada: A discrete event approach to the control of robotic assembly tasks, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1993) pp. 331–336
- 26.75 B.J. McCarragher, H. Asada: The discrete event modeling and trajectory planning of robotic assembly tasks, ASME J. Dyn. Syst. Meas. Contr. **117**, 394–400 (1995)
- 26.76 R. Suárez, L. Basañez, J. Rosell: Using configuration and force sensing in assembly task planning and

- execution, *Proc. Int. Symp. Assem. Task Plan.* (1995) pp. 273–279
- 26.77 H. Hirukawa, Y. Papegay, T. Matsui: A motion planning algorithm for convex polyhedra in contact under translation and rotation, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (San Diego, 1994) pp. 3020–3027
- 26.78 X. Ji, J. Xiao: Planning motion compliant to complex contact states, *Int. J. Robot. Res.* **20**(6), 446–465 (2001)
- 26.79 L.E. Kavraki, P. Svestka, J.C. Latombe, M. Overmars: Probabilistic roadmaps for path planning in high-dimensional configuration spaces, *IEEE Trans. Robot. Autom.* **12**(4), 566–580 (1996)
- 26.80 B. Hannaford, P. Lee: Hidden Markov model analysis of force/torque information in telemanipulation, *Int. J. Robot. Res.* **10**(5), 528–539 (1991)
- 26.81 G.E. Hovland, B.J. McCarragher: Hidden Markov models as a process monitor in robotic assembly, *Int. J. Robot. Res.* **17**(2), 153–168 (1998)
- 26.82 T. Takahashi, H. Ogata, S. Muto: A method for analyzing human assembly operations for use in automatically generating robot commands, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol. 2 (Atlanta 1993) pp. 695–700
- 26.83 P. Sikka, B.J. McCarragher: Rule-based contact monitoring using examples obtained by task demonstration, *Proc. 15th Int. Jt. Conf. Artif. Intell.* (Nagoya 1997) pp. 514–521
- 26.84 E. Cervera, A. Del Pobil, E. Marta, M. Serna: Perception-based learning for motion in contact in task planning, *J. Intell. Robot. Syst.* **17**(3), 283–308 (1996)
- 26.85 L.M. Brignone, M. Howarth: A geometrically validated approach to autonomous robotic assembly, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* (Lausanne 2002) pp. 1626–1631
- 26.86 M. Nuttin, J. Rosell, R. Suárez, H. Van Brussel, L. Basañez, J. Hao: Learning approaches to contact estimation in assembly tasks with robots, *Proc. 3rd Eur. Workshop Learn. Robot.* (Heraklion 1995)
- 26.87 L.J. Everett, R. Ravari, R.A. Volz, M. Skubic: Generalized recognition of single-ended contact formations, *IEEE Trans. Robot. Autom.* **15**(5), 829–836 (1999)
- 26.88 M. Skubic, R.A. Volz: Identifying single-ended contact formations from force sensor patterns, *IEEE Trans. Robot. Autom.* **16**(5), 597–603 (2000)
- 26.89 S. Hirai, H. Asada: Kinematics and statics of manipulation using the theory of polyhedral convex cones, *Int. J. Robot. Res.* **12**(5), 434–447 (1993)
- 26.90 H. Hirukawa, T. Matsui, K. Takase: Automatic determination of possible velocity and applicable force of frictionless objects in contact from a geometric model, *IEEE Trans. Robot. Autom.* **10**(3), 309–322 (1994)
- 26.91 B.J. McCarragher, H. Asada: Qualitative template matching using dynamic process models for state transition recognition of robotic assembly, *ASME J. Dyn. Syst. Meas. Contr.* **115**(2), 261–269 (1993)
- 26.92 T.M. Schulteis, P.E. Dupont, P.A. Millman, R.D. Howe: Automatic identification of remote environments, *Proc. ASME Dyn. Syst. Contr. Div.* (Atlanta 1996) pp. 451–458
- 26.93 A.O. Farahat, B.S. Graves, J.C. Trinkle: Identifying contact formations in the presence of uncertainty, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* (Pittsburg 1995) pp. 59–64
- 26.94 J. Xiao, L. Zhang: Contact constraint analysis and determination of geometrically valid contact formations from possible contact primitives, *IEEE Trans. Robot. Autom.* **13**(3), 456–466 (1997)
- 26.95 H. Mosemann, T. Bierwirth, F.M. Wahl, S. Stoeter: Generating polyhedral convex cones from contact graphs for the identification of assembly process states, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (2000) pp. 744–749
- 26.96 J. Xiao, L. Zhang: Towards obtaining all possible contacts – growing a polyhedron by its location uncertainty, *IEEE Trans. Robot. Autom.* **12**(4), 553–565 (1996)
- 26.97 M. Spreng: A probabilistic method to analyze ambiguous contact situations, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Atlanta 1993) pp. 543–548
- 26.98 N. Mimura, Y. Funahashi: Parameter identification of contact conditions by active force sensing, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (San Diego 1994) pp. 2645–2650
- 26.99 B. Eberman: A model-based approach to Cartesian manipulation contact sensing, *Int. J. Robot. Res.* **16**(4), 508–528 (1997)
- 26.100 T. Debus, P. Dupont, R. Howe: Contact state estimation using multiple model estimation and hidden Markov model, *Int. J. Robot. Res.* **23**(4–5), 399–413 (2004)
- 26.101 J. De Geeter, H. Van Brussel, J. De Schutter, M. Decréton: Recognizing and locating objects with local sensors, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Minneapolis 1996) pp. 3478–3483
- 26.102 J. De Schutter, H. Bruyninckx, S. Dutré, J. De Geeter, J. Katupitiya, S. Demey, T. Lefebvre: Estimating first-order geometric parameters and monitoring contact transitions during force-controlled compliant motions, *Int. J. Robot. Res.* **18**(12), 1161–1184 (1999)
- 26.103 T. Lefebvre, H. Bruyninckx, J. De Schutter: Polyhedral contact formation identification for autonomous compliant motion: exact nonlinear bayesian filtering, *IEEE Trans. Robot.* **21**(1), 124–129 (2005)
- 26.104 T. Lefebvre, H. Bruyninckx, J. De Schutter: Online statistical model recognition and state estimation for autonomous compliant motion systems, *IEEE Trans. Syst. Man Cybern. Part C Special Issue on “Pattern Recognition for Autonomous Manipulation in Robotic Systems”* **35**(1), 16–29 (2005)

- 26.105 K. Gadeyne, T. Lefebvre, H. Bruyninckx: Bayesian hybrid model-state estimation applied to simultaneous contact formation recognition and geometrical parameter estimation, *Int. J. Robot. Res.* **24**(8), 615–630 (2005)
- 26.106 K. Kitagaki, T. Ogasawara, T. Suehiro: Methods to detect contact state by force sensing in an edge mating task, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Atlanta 1993) pp. 701–706
- 26.107 T. Lefebvre, H. Bruyninckx, J. De Schutter: Task planning with active sensing for autonomous compliant motion, *Int. J. Robot. Res.* **24**(1), 61–82 (2005)
- 26.108 T. Debus, P. Dupont, R. Howe: Distinguishability and identifiability testing of contact state models, *Adv. Robot.* **19**(5), 545–566 (2005)
- 26.109 J. Park, R. Cortesao, O. Khatib: Multi-contact compliant motion control for robotic manipulators, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (New Orleans 2004) pp. 4789–4794
- 26.110 W. Meeussen, J. De Schutter, H. Bruyninckx, J. Xiao, E. Staffetti: Integration of planning and execution in force controlled compliant motion, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* (2005)
- 26.111 D.E. Koditschek: Exact robot navigation by means of potential functions: some topological considerations, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Raleigh 1987) pp. 1–6
- 26.112 E. Rimon, D.E. Koditschek: Exact robot navigation using artificial potential fields, *IEEE Trans. Robot. Autom.* **8**(5), 501–518 (1992)
- 26.113 E. Rimon, D.E. Koditschek: The construction of analytic diffeomorphisms for exact robot navigation on star worlds, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Scottsdale 1989) pp. 21–26
- 26.114 J. Barraquand, J.-C. Latombe: Robot motion planning: a distributed representation approach, *Int. J. Robot. Res.* **10**(6), 628–649 (1991)
- 26.115 C.I. Connolly, J.B. Burns, R. Weiss: Path planning using Laplace's equation, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Cincinnati 1990) pp. 2102–2106
- 26.116 C.I. Connolly, R.A. Grupen: One the applications of harmonic functions to robotics, *J. Robot. Syst.* **10**(7), 931–946 (1993)
- 26.117 S.H.J. Feder, E.J.-J. Slotine: Real-time path planning using harmonic potentials in dynamic environments, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Albuquerque 1997) pp. 811–874
- 26.118 J.-O. Kim, P. Khosla: Real-time obstacle avoidance using harmonic potential functions, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Sacramento 1991) pp. 790–796
- 26.119 K. Sato: Collision avoidance in multi-dimensional space using Laplace potential, *Proc. 15th Conf. Robot. Soc. Jpn.* (1987), 155–156
- 26.120 S.M. LaValle, P. Konkimalla: Algorithms for computing numerical optimal feedback motion strategies, *Int. J. Robot. Res.* **20**(9), 729–752 (2001)
- 26.121 A. van der Schaft, H. Schumacher: *An Introduction to Hybrid Dynamical Systems* (Springer, Berlin, Heidelberg 2000)
- 26.122 W. Choi, J.-C. Latombe: A reactive architecture for planning and executing robot motions with incomplete knowledge, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Vol. 1 (Osaka 1991) pp. 24–29
- 26.123 D. Conner, H. Choset, A. Rizzi: Integrated planning and control for convex-bodied nonholonomic systems using local feedback control policies, *Proc. Robot.: Sci. Syst.* (Philadelphia 2006)
- 26.124 D.C. Conner, A.A. Rizzi, H. Choset: Composition of local potential functions for global robot control and navigation, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* (Las Vegas 2003) pp. 3546–3551
- 26.125 S.R. Lindemann, S.M. LaValle: Smooth feedback for car-like vehicles in polygonal environments, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Rome 2007)
- 26.126 L. Yang, S.M. LaValle: The sampling-based neighborhood graph: a framework for planning and executing feedback motion strategies, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Taipei 2003)
- 26.127 C. Belta, V. Isler, G.J. Pappas: Discrete abstractions for robot motion planning and control in polygonal environments, *IEEE Trans. Robot. Autom.* **21**(5), 864–871 (2005)
- 26.128 R.R. Burridge, A.A. Rizzi, D.E. Koditschek: Sequential composition of dynamically dexterous robot behaviors, *Int. J. Robot. Res.* **18**(6), 534–555 (1999)
- 26.129 S.R. Lindemann, S.M. LaValle: Computing smooth feedback plans over cylindrical algebraic decompositions, *Proc. Robot.: Sci. Syst. (RSS)* (Philadelphia 2006)
- 26.130 Y. Yang, O. Brock: Elastic roadmaps: globally task-consistent motion for autonomous mobile manipulation, *Proc. Robot.: Sci. Syst. (RSS)* (Philadelphia 2006)
- 26.131 S. Quinlan, O. Khatib: Elastic bands: connecting path planning and control, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol. 2 (Atlanta 1993) pp. 802–807
- 26.132 M. Khatib, H. Jaouni, R. Chatila, J.-P. Laumond: How to implement dynamic paths, *Proc. Int. Symp. Exp. Robot.* (1997) pp. 225–236, Preprints
- 26.133 O. Brock, O. Khatib: Elastic strips: a framework for motion generation in human environments, *Int. J. Robot. Res.* **21**(12), 1031–1052 (2002)
- 26.134 M.H. Raibert, J.J. Craig: Hybrid position/force control of manipulators, *J. Dyn. Syst. Meas. Contr.* **103**(2), 126–133 (1981)
- 26.135 R. Featherstone, S. Sonck, O. Khatib: A general contact model for dynamically-decoupled force/motion control, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (Detroit 1999) pp. 3281–3286
- 26.136 J. Park, R. Cortes ao, O. Khatib: Multi-contact compliant motion control for robotic manipulators, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)* (New Orleans 2004) pp. 4789–4794

- 26.137 J. Park, O. Khatib: Multi-link multi-contact force control for manipulators, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Barcelona 2005)
- 26.138 D.C. Ruspini, O. Khatib: A framework for multi-contact multi-body dynamic simulation and haptic display, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (Takamatsu 2000) pp.1322–1327
- 26.139 K.-S. Chang, R. Holmberg, O. Khatib: The augmented object model: cooperative manipulation and parallel mechanism dynamics, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (San Francisco 2000) pp.470–475
- 26.140 O. Khatib: Object Manipulation in a Multi-Effector Robot System. In: *Robotics Research 4*, ed. by R. Bolles, B. Roth (MIT Press, Cambridge 1988) pp.137–144
- 26.141 D. Williams, O. Khatib: The virtual linkage: a model for internal forces in multi-grasp manipulation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Vol.1 (Atlanta 1993) pp.1030–1035
- 26.142 J.A. Adams, R. Bajcsy, J. Kosecka, V. Kuma, R. Mandelbaum, M. Mintz, R. Paul, C. Wang, Y. Yamamoto, X. Yun: Cooperative material handling by human and robotic agents: module development and system synthesis, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (Pittsburgh 1995) pp.200–205
- 26.143 S. Hayati: Hybrid position/force control of multi-arm cooperating robots, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (San Francisco 1986) pp.82–89
- 26.144 D. Jung, G. Cheng, A. Zelinsky: Experiments in realizing cooperation between autonomous mobile robots, Proc. Int. Symp. Exp. Robot. (1997) pp.513–524
- 26.145 T.-J. Tarn, A.K. Bejczy, X. Yun: Design of dynamic control of two cooperating robot arms: Closed chain formulation, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1987) pp.7–13
- 26.146 M. Uchiyama, P. Dauchez: A symmetric hybrid position/force control scheme for the coordination of two robots, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Philadelphia 1988) pp.350–356
- 26.147 X. Yun, V.R. Kumar: An approach to simultaneous control of trajectory and interaction forces in dual-arm configurations, IEEE Trans. Robot. Autom. 7(5), 618–625 (1991)
- 26.148 Y.F. Zheng, J.Y.S. Luh: Joint torques for control of two coordinated moving robots, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (San Francisco 1986) pp.1375–1380
- 26.149 J. Russakow, O. Khatib, S.M. Rock: Extended operational space formation for serial-to-parallel chain (branching) manipulators, Proc. IEEE Int. Conf. Robot. Autom. (ICRA), Vol.1 (Nagoya 1995) pp.1056–1061
- 26.150 K.-S. Chang, O. Khatib: Operational space dynamics: efficient algorithms for modelling and control of branching mechanisms, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (San Francisco 2000) pp.850–856
- 26.151 K. Kreutz-Delgado, A. Jain, G. Rodriguez: Recursive formulation of operational space control, Int. J. Robot. Res. 11(4), 320–328 (1992)
- 26.152 B. Bayle, J.-Y. Fourquet, M. Renaud: A coordination strategy for mobile manipulation, Proc. Int. Conf. Intell. Auton. Syst. (Venice 2000) pp.981–988
- 26.153 B. Bayle, J.-Y. Fourquet, M. Renaud: Generalized path generation for a mobile manipulator, Proc. Int. Conf. Mech. Des. Prod. (Cairo 2000) pp.57–66
- 26.154 B. Bayle, J.-Y. Fourquet, M. Renaud: Using manipulability with nonholonomic mobile manipulators, Proc. Int. Conf. Field Serv. Robot. (Helsinki 2001) pp.343–348
- 26.155 J. Peters, S. Schaal: Reinforcement learning for operational space control, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Rome 2007)
- 26.156 T. Simeon, J. Cortes, A. Sahbani, J.P. Laumond: A manipulation planner for pick and place operations under continuous grasps and placements, Proc. IEEE Int. Conf. Robot. Autom. (2002)
- 26.157 A. Miller, P. Allen: Examples of 3D grasp quality computations, Robot. Autom. 1999. Proc. 1999 IEEE Int. Conf. on, Vol.2 (1999)
- 26.158 A.T. Miller: GrasPlt: A Versatile Simulator for Robotic Grasping. Ph.D. Thesis (Department of Computer Science, Columbia University 2001)
- 26.159 G. Wilfong: Motion planning in the presence of movable obstacles, Proc. ACM Symp. Computat. Geom. (1988) pp.279–288
- 26.160 M. Erdmann, T. Lozano-Perez: On multiple moving objects, IEEE Int. Conf. Robot. Autom. (San Francisco 1986) pp.1419–1424
- 26.161 P.C. Chen, Y.K. Hwang: Pratical path planning among movable obstacles, Proc. IEEE Int. Conf. Robot. Autom. (1991) pp.444–449
- 26.162 M. Stilman, J.J. Kuffner: Navigation among movable obstacles: real-time reasoning in complex environments, Int. J. Human Robot. 2(4), 1–24 (2005)
- 26.163 M. Stilman, J.-U. Shamburek, J.J. Kuffner, T. Asfour: Manipulation planning among movable obstacles, Proc. IEEE Int. Conf. Robot. Autom. (ICRA'07) (2007)
- 26.164 D. Nieuwenhuisen, A.F. van der Stappen, M.H. Overmars: An effective framework for path planning amidst movable obstacles, Workshop Algorithm. Fund. Robot. (2006)
- 26.165 K.M. Lynch, M.T. Mason: Stable pushing: mechanics, controllability, and planning, Int. J. Robot. Res. 15(6), 533–556 (1996)
- 26.166 P. Tang, J. Xiao: Generation of point-contact state space between strictly curved objects. In: *Robotics Science and Systems II*, ed. by G.S. Sukhatme, S. Schaal, W. Burgard, D. Fox (MIT Press, Cambridge 2007) pp.239–246
- 26.167 H. Nakagaki, K. Kitagaki, T. Ogasawara, H. Tsukune: Study of deformation and insertion tasks of a flex-

- ible wire, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1997) pp. 2397–2402
- 26.168 W. Kraus Jr., B.J. McCarragher: Case studies in the manipulation of flexible parts using a hybrid position/force approach, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1997) pp. 367–372
- 26.169 J.Y. Kim, D.J. Kang, H.S. Cho: A flexible parts assembly algorithm based on a visual sensing system, Proc. Int. Symp. Assem. Task Plan. (2001) pp. 417–422
- 26.170 B.J. Unger, A. Nocolaidis, P.J. Berkelman, A. Thompson, R.L. Klatzky, R.L. Hollis: Comparison of 3-D haptic peg-in-hole tasks in real and virtual environments, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (2001) pp. 1751–1756

27. Contact Modeling and Manipulation

Imin Kao, Kevin Lynch, Joel W. Burdick

Robotic manipulators use contact forces to grasp and manipulate objects in their environments. Fixtures rely on contacts to immobilize workpieces. Mobile robots and humanoids use wheels or feet to generate the contact forces that allow them to locomote. Modeling of the *contact interface*, therefore, is fundamental to analysis, design, planning, and control of many robotic tasks.

This chapter presents an overview of the modeling of contact interfaces, with a particular focus on their use in manipulation tasks, including graspless or *nonprehensile* manipulation modes such as pushing. Analysis and design of grasps and fixtures also depends on contact modeling, and these are discussed in more detail in Chap. 28. Sections 27.2–27.5 focus on rigid-body models of contact. Section 27.2 describes the kinematic constraints caused by contact, and Sect. 27.3 describes the contact forces that may arise with Coulomb friction. Section 27.4 provides examples of analysis of multicontact manipulation tasks with rigid bodies and Coulomb friction. Section 27.5 extends the analysis to manipulation by pushing. Section 27.6 introduces modeling of contact interfaces, kinematic duality, and pressure distribution. Section 27.7 describes the concept of the friction limit surface and illustrates it with an example demonstrating the construction of a limit surface for a soft contact. Finally, Sect. 27.8 discusses how these more accurate models can be used in fixture analysis and design.

27.1	Overview	647
27.1.1	Choosing a Contact Model	648
27.1.2	Grasp/Manipulation Analysis	648
27.2	Kinematics of Rigid-Body Contact	648
27.2.1	Contact Constraints.....	649
27.2.2	Collections of Parts	650
27.2.3	Graphical Planar Methods	651
27.3	Forces and Friction	652
27.3.1	Graphical Planar Methods	653
27.3.2	Duality of Contact Wrenches and Twist Freedoms.....	653
27.4	Rigid-Body Mechanics with Friction	655
27.4.1	Complementarity	655
27.4.2	Quasistatic Assumption	656
27.4.3	Examples	656
27.5	Pushing Manipulation	658
27.6	Contact Interfaces and Modeling	659
27.6.1	Modeling of Contact Interface	659
27.6.2	Pressure Distribution at Contacts....	660
27.7	Friction Limit Surface	661
27.7.1	The Friction Limit Surface at a Soft Contact Interface	662
27.7.2	Example of Constructing a Friction Limit Surface	663
27.8	Contacts in Grasping and Fixture Designs	664
27.8.1	Contact Stiffness of Soft Fingers.....	664
27.8.2	Application of Soft Contact Theory to Fixture Design.....	665
27.9	Conclusions and Further Reading	666
	References	666

27.1 Overview

A contact model characterizes both the forces that can be transmitted through the contact as well as the allowed relative motions of the contacting bodies. These char-

acteristics are determined by the geometry of the contacting surfaces and the material properties of the parts, which dictate friction and possible contact deformation.

27.1.1 Choosing a Contact Model

The choice of a contact model largely depends upon the application or analysis that must be carried out. When appropriate analytical models are used, one can determine if a manipulation plan or fixture design meets desired functional requirements within the model's limits.

Rigid-Body Models

Many approaches to manipulation, grasp, and fixture analysis are based on rigid-body models. In the rigid-body model, no deformations are allowed at the points or surfaces of contact between two bodies. Instead, contact forces arise from two sources: the constraint of incompressibility and impenetrability between the rigid bodies, and surface frictional forces. Rigid-body models are straightforward to use, lead to computationally efficient planning algorithms, and are compatible with solid-modeling software systems. Rigid-body models are often appropriate for answering qualitative questions such as *will this fixture be able to hold my workpiece?* and for problems involving stiff parts with low to moderate contact forces.

Rigid-body models are not capable of describing the full range of contact phenomena, however; for example, rigid-body models cannot predict the individual contact forces of a multiple-contact fixture (the static indeterminacy problem [27.1, 2]). Furthermore, workpieces held in fixtures experience non-negligible deformations in many high-force manufacturing operations [27.3–6]. These deformations, which crucially impact machining accuracy, cannot be determined from rigid-body models. Also, a rigid-body model augmented with a Coulomb friction model can lead to mechanics problems that have no solution or multiple solutions [27.7–15]. To overcome the limitations inherent in the rigid-body model, one must introduce compliance into the contact model.

Compliant Models

A *compliant* contact deforms under the influence of applied forces. The forces of interaction at the contact are derived from the *compliance* or *stiffness* model. While

compliant contact models are typically more complicated, they have several advantages: they overcome the static indeterminacy inherent to rigid-body models and they predict the deformations of grasped or fixtured parts during loading.

A detailed model of the deformations of real materials can be quite complex. Consequently, for analysis we often introduce *lumped-parameter* or *reduced-order* compliance models having a limited number of variables. In this chapter we summarize a reduced-order *quasi-rigid-body* approach to modeling that can model a variety of compliant materials in a way that is consistent with both the solid mechanics literature and conventional robot analysis and planning paradigms.

Finally, three-dimensional finite-element models [27.16–18] or similar ideas [27.19, 20] can be used to analyze workpiece deformations and stresses in fixtures. While accurate, these numerical approaches have some drawbacks, for example, the grasp stiffness matrix can only be found through difficult numerical procedures. Stiffness matrices are often needed to compute quality measures that are the basis for optimal grasping plans or fixturing designs [27.21, 22]. Thus, these numerical approaches are better suited for verifying final fixture designs.

27.1.2 Grasp/Manipulation Analysis

Once a contact model has been chosen, we can use it to analyze tasks involving multiple contacts. If a part is subject to multiple contacts, the kinematic constraints and force freedoms due to the individual contacts must be combined. This combined analysis facilitates manipulation planning – choosing the contact locations, and possibly the motions or forces applied by those contacts, to achieve the desired behavior of the part. A prime example is the grasping or fixturing problem: choosing contact locations, and possibly contact forces, to prevent motion of a part in the face of external disturbances. This well-studied topic is discussed in greater detail in Chap. 28. Other examples include problems of partial constraint, such as pushing a part or inserting a peg into a hole.

27.2 Kinematics of Rigid-Body Contact

Contact kinematics is the study of how two or more parts can move relative to each other while respecting the rigid-body impenetrability constraint. It also classifies motion in contact as either rolling or slipping.

Consider two rigid bodies whose position and orientation (configuration) is given by the local coordinate column vectors \mathbf{q}_1 and \mathbf{q}_2 , respectively. Writing the composite configuration as $\mathbf{q} = (\mathbf{q}_1^\top, \mathbf{q}_2^\top)^\top$, we define

a *distance* function $d(\mathbf{q})$ between the parts that is positive when they are separated, zero when they are touching, and negative when they are in penetration. When $d(\mathbf{q}) > 0$, there are no constraints on the motions of the parts. When the parts are in contact ($d(\mathbf{q}) = 0$), we look at the time derivatives \dot{d} , \ddot{d} , etc., to determine if the parts stay in contact or break apart as they follow a trajectory $\mathbf{q}(t)$. This can be determined by the following table of possibilities:

d	\dot{d}	\ddot{d}	...
> 0			no contact,
< 0			infeasible (penetration),
$= 0 > 0$			breaking contact,
$= 0 < 0$			infeasible (penetration),
$= 0 = 0 > 0$			breaking contact,
$= 0 = 0 < 0$			infeasible (penetration),
etc.			

The contact is maintained only if all time derivatives are zero.

The first two time derivatives are written

$$\dot{d} = \left(\frac{\partial d}{\partial \mathbf{q}} \right)^\top \dot{\mathbf{q}}, \quad (27.1)$$

$$\ddot{d} = \dot{\mathbf{q}}^\top \frac{\partial^2 d}{\partial \mathbf{q}^2} \dot{\mathbf{q}} + \left(\frac{\partial d}{\partial \mathbf{q}} \right)^\top \ddot{\mathbf{q}}. \quad (27.2)$$

The terms $\partial d / \partial \mathbf{q}$ and $\partial^2 d / \partial \mathbf{q}^2$ carry information about the local contact geometry. The former corresponds to the contact normal, while the latter corresponds to the relative curvature of the parts at the contact.

If contact is maintained, we can classify the contact as slipping or rolling. Analogous to the table above, the contact is rolling if and only if there is zero relative tangential velocity, acceleration, etc., between the contact points on the parts. If the relative tangential velocity is nonzero, the parts are slipping; if the relative velocity is zero but relative tangential acceleration or (higher-order derivatives) is not, slipping is incipient.

In this section we focus on a *first-order* analysis of contact kinematics. A first-order analysis concludes that contact is maintained if $d(\mathbf{q}) = 0$ and $\dot{d} = 0$. This local linearization of contact kinematics focuses on the velocity $\dot{\mathbf{q}}$ and the contact normal in $\partial d / \partial \mathbf{q}$; higher-order spatial derivatives of the contact geometry (curvature, etc.) are not considered. While this is a good starting point, it may occasionally lead to erroneous conclusions. For example, *Rimon* and *Burdick* [27.23–25] showed that a first-order analysis may incorrectly predict mobil-

ity of a part in a fixture when a second-order analysis shows that it is in fact completely constrained.

Analysis of the kinematics of rolling contact of parameterized surfaces can be found in [27.26]; see also [27.27–30].

27.2.1 Contact Constraints

As described in Chap. 1, a rigid body in space has six degrees of freedom, specified by the location of the origin of a coordinate frame P affixed to the part and the orientation of this coordinate frame relative to an inertial frame O fixed in the world. Let ${}^O\mathbf{p}_P \in \mathbb{R}^3$ be the position of the part center of mass and ${}^O\mathbf{R}_P \in SO(3)$ be the rotation matrix describing the orientation of the part relative to O . The spatial velocity of the part can be written as $\mathbf{t} \in \mathbb{R}^6$, sometimes called a *twist*,

$$\mathbf{t} = \left(\boldsymbol{\omega}^\top, \mathbf{v}^\top \right)^\top,$$

where $\boldsymbol{\omega} = (\omega_x, \omega_y, \omega_z)^\top$ and $\mathbf{v} = (v_x, v_y, v_z)^\top$ give the angular velocity and linear velocity of P in the world frame O , respectively, such that $\boldsymbol{\omega}$ satisfies

$${}^O\dot{\mathbf{R}}_P = \boldsymbol{\omega} \times {}^O\mathbf{R}_P$$

and \mathbf{v} satisfies

$$\mathbf{v} = {}^O\dot{\mathbf{p}}_P - \boldsymbol{\omega} \times {}^O\mathbf{p}_P.$$

It is worth taking a moment to really understand the spatial velocity of a body. It consists of the body's angular velocity expressed in the world frame O , along with the linear velocity of a point as if it were rigidly attached to the body but currently at the origin of the world frame. This point need not be physically on the body. In other words, \mathbf{v} is not simply ${}^O\dot{\mathbf{p}}_P$. This notation will simplify the following expressions, where all velocities and forces will be expressed in the common world frame O . (Be aware that twists are sometimes defined in a body frame instead.)

A point contact acting on the part provides a unilateral constraint which prevents the part from locally moving against the contact normal. Let \mathbf{x} be the location of the contact in O . The linear velocity of the point on the part in contact is

$$\mathbf{v}_C = \mathbf{v} + \boldsymbol{\omega} \times \mathbf{x}.$$

(We drop the pre-superscripts O for simplicity; for example, ${}^O\mathbf{x}$ is written simply as \mathbf{x} .) Let $\hat{\mathbf{u}}$ be the unit vector normal pointing into the part (Fig. 27.1). The first-order condition that the part not move into the unilateral

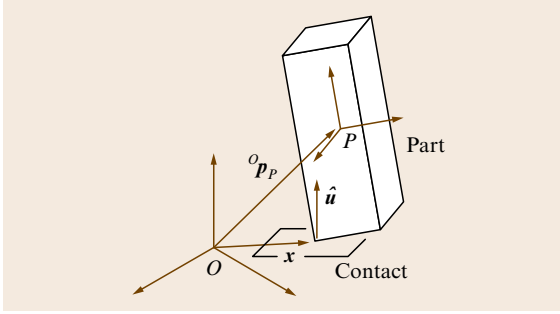


Fig. 27.1 Notation for a part in contact with a manipulator or the environment

constraint can be written

$$\mathbf{v}_C^\top \hat{\mathbf{u}} = (\mathbf{v} + \boldsymbol{\omega} \times \mathbf{x})^\top \hat{\mathbf{u}} \geq 0. \quad (27.3)$$

In other words, the velocity of the part at C cannot have a component in the opposite direction of the contact normal. To write this another way, define a generalized force or *wrench* \mathbf{w} consisting of the torque \mathbf{m} and force \mathbf{f} acting on the part for a unit force along the contact normal:

$$\mathbf{w} = (\mathbf{m}^\top, \mathbf{f}^\top)^\top = [(\mathbf{x} \times \hat{\mathbf{u}})^\top, \hat{\mathbf{u}}^\top]^\top.$$

Then (27.3) can be rewritten as

$$\mathbf{t}^\top \mathbf{w} \geq 0. \quad (27.4)$$

If the external constraint point is moving with linear velocity \mathbf{v}_{ext} , (27.4) changes to

$$\mathbf{t}^\top \mathbf{w} \geq \mathbf{v}_{\text{ext}}^\top \hat{\mathbf{u}}, \quad (27.5)$$

which reduces to (27.4) if the constraint is stationary.

Each inequality of the form (27.5) constrains the velocity of the part to a half-space of its six-dimensional velocity space bounded by the hyperplane $\mathbf{t}^\top \mathbf{w} = \mathbf{v}_{\text{ext}}^\top \hat{\mathbf{u}}$. Unioning the set of all constraints, we get a convex polyhedral set of feasible part velocities. A constraint is redundant if its half-space constraint does not change the feasible velocity polyhedron. For a given twist \mathbf{t} , a constraint is active if

$$\mathbf{t}^\top \mathbf{w} = \mathbf{v}_{\text{ext}}^\top \hat{\mathbf{u}}; \quad (27.6)$$

otherwise the part is breaking contact at that point. In general, the feasible velocity polyhedron for a part can consist of a six-dimensional interior (where no contact constraint is active), five-dimensional hyperfaces, four-dimensional hyperfaces, and so on, down to one-dimensional edges and zero-dimensional points. A part

velocity on an n -dimensional facet of the velocity polyhedron indicates that $6 - n$ independent (nonredundant) constraints are active.

If all of the constraints are stationary ($\mathbf{v}_{\text{ext}} = 0$), then each constraining half-plane defined by (27.5) passes through the origin of the velocity space, and the feasible velocity set becomes a cone rooted at the origin. Let \mathbf{w}_i be the constraint wrench of stationary contact i . Then the feasible velocity cone is

$$V = \{\mathbf{t} \mid \mathbf{t}^\top \mathbf{w}_i \geq 0 \quad \forall i\}.$$

If the \mathbf{w}_i span the six-dimensional generalized force space, or, equivalently, the convex hull of the \mathbf{w}_i contains the origin in the interior, then the stationary contacts completely constrain the motion of the part, and we have form closure, as discussed in Chap. 28.

In the discussion above, each constraint (27.5) divides the part velocity space into three categories: a hyperplane of velocities that maintain contact, a half-space of velocities that separate the parts, and a half-space of velocities that cause the parts to penetrate. Velocities where the contact is maintained can be further broken down into two categories: velocities where the part slips over the contact constraint, and velocities where the part sticks or rolls on the constraint. In the latter case, the part velocity satisfies the three equations

$$\mathbf{v} + \boldsymbol{\omega} \times \mathbf{x} = \mathbf{v}_{\text{ext}}. \quad (27.7)$$

Now we can give each point contact i a label m_i corresponding to the type of contact, called the contact label: \mathbf{b} if the contact is breaking, \mathbf{f} if the contact is fixed (including rolling), and \mathbf{s} if the contact is slipping, i. e., (27.6) is satisfied but (27.7) is not. The *contact mode* for the entire system can be written as the concatenation of the contact labels at the k contacts, $m_1 m_2 \dots m_k$.

27.2.2 Collections of Parts

The discussion above can be generalized to find the feasible velocities of multiple parts in contact. If parts i and j make contact at a point \mathbf{x} , where $\hat{\mathbf{u}}_i$ points into part i and $\mathbf{w}_i = [(\mathbf{x} \times \hat{\mathbf{u}}_i)^\top, \hat{\mathbf{u}}_i^\top]^\top$, then their spatial velocities \mathbf{t}_i and \mathbf{t}_j must satisfy the constraint

$$(\mathbf{t}_i - \mathbf{t}_j)^\top \mathbf{w}_i \geq 0 \quad (27.8)$$

to avoid penetration. This is a homogeneous half-space constraint in the composite $(\mathbf{t}_i, \mathbf{t}_j)$ velocity space. In an assembly of multiple parts, each pairwise contact contributes another constraint in the composite part velocity

space, and the result is a polyhedral convex cone of kinematically feasible velocities rooted at the origin of the composite velocity space. The contact mode for the entire assembly is the concatenation of the contact labels at each contact in the assembly.

If there are moving contacts whose motion is prescribed, e.g., robot fingers, the constraints on the motion of the remaining parts will no longer be homogeneous. As a result, the convex polyhedral feasible velocity space is no longer a cone rooted at the origin.

27.2.3 Graphical Planar Methods

When a part is confined to move in the x - y -plane, the twist \mathbf{t} reduces to $\mathbf{t} = (\omega_x, \omega_y, \omega_z, v_x, v_y, v_z)^\top = (0, 0, \omega_z, v_x, v_y, 0)^\top$. The point $(-v_y/\omega_z, v_x/\omega_z)$ is called the center of rotation (CoR) in the projective plane, and we can represent any planar twist by its CoR and rotational velocity ω_z . (Note that the case $\omega_z = 0$ must be treated with care, as it corresponds to a CoR at infinity.) This is sometimes useful for graphical purposes: for a single part constrained by stationary fixtures, at least, we can easily draw the feasible twist cone as CoRs [27.14, 31].

As an example, Fig. 27.2a shows a planar part standing on a table and being contacted by a robot finger. This finger is currently stationary, but we will later set it in motion. The finger defines one constraint on the part's motion and the table defines two more. (Note that contact points in the interior of the edge between the part and the table provide redundant kinematic constraints.) The constraint wrenches can be written

$$\begin{aligned}\mathbf{w}_1 &= (0, 0, -1, 0, 1, 0)^\top, \\ \mathbf{w}_2 &= (0, 0, 1, 0, 1, 0)^\top, \\ \mathbf{w}_3 &= (0, 0, 1, -1, 0, 0)^\top.\end{aligned}$$

For a stationary finger, the kinematic constraints yield the feasible twist cone shown in Fig. 27.2b. This region can also be easily visualized in the plane by using the following method: at each contact, draw the contact normal line. Label all points on the normal \pm , points to the left of the inward normal $+$, and points to the right $-$. For each contact constraint, all the points labeled $+$ can serve as CoRs with positive angular velocity, and all the points labeled $-$ can serve as CoRs with negative angular velocity, without violating the contact constraint. After doing this for all the contact normals, keep only the CoRs that are consistently labeled. These CoRs are a planar representation of the feasible twist cone (Fig. 27.2c).

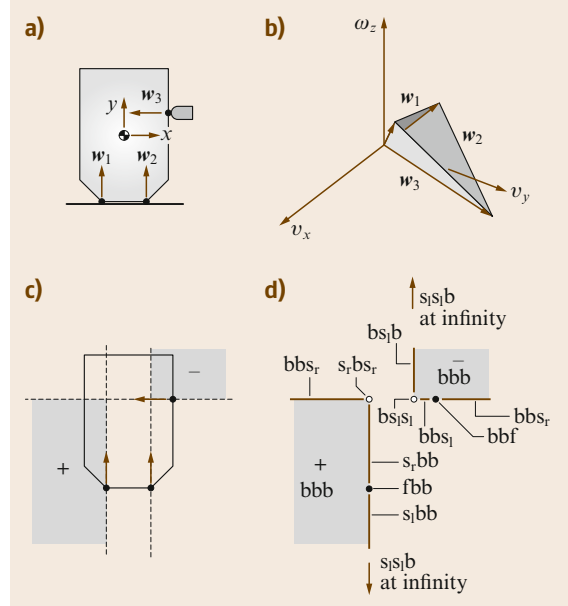


Fig. 27.2 (a) A part resting on a table and the three contact constraints. (b) The twist cone satisfying the contact constraints. The contact normals are shown with their associated constraint planes. (c) The equivalent representation as CoRs, shown in grey. Note that the lines that extend off to the *left* and to the *bottom* wrap around at infinity and come back in from the *right* and the *top*, respectively, so this COR region should be interpreted as a single connected convex region, equivalent to the twist cone representation in (b). (d) The contact modes assigned to each feasible motion. The zero velocity contact mode is fff

We can refine this method by assigning contact modes to each feasible CoR. For each contact normal, label the CoR at the contact point f for fixed, other CoRs on the normal line s for slipping, and all other CoRs b for breaking contact. The concatenation of these labels gives the part's contact mode for a particular part motion. In the planar case, the label s on the contact normal line can be further refined into s_r or s_l , indicating whether the part is slipping right or left relative to the constraint. An s CoR labeled $+$ above the contact (in the direction of the contact normal) or $-$ below the contact should be relabeled s_r , and an s CoR labeled $-$ above the contact or $+$ below should be relabeled s_l (Fig. 27.2d).

This method can be used readily to determine if the part is in form closure. If there is no CoR labeled consistently, then the feasible velocity cone consists of only

the zero velocity point, and the part is immobilized by the stationary contacts. This method also makes it clear that at least four contacts are necessary to immobilize the part by the first-order analysis (see also Chap. 28). This is a weakness of the first-order analysis – curvature effects can be used to immobilize a part with three

or even two contacts [27.24]. This weakness can also be seen in Fig. 27.2d. A pure rotation about the CoR labeled $\{+, s_r, b, s_r\}$ is actually not feasible, but it would be if the part had a small radius of curvature at the contact with the finger. The first-order analysis ignores this curvature.

27.3 Forces and Friction

A commonly used model of friction in robotic manipulation is Coulombs law [27.32]. This experimental law states that the friction force magnitude f_t in the tangent plane at the contact interface is related to the normal force magnitude f_n by $f_t \leq \mu f_n$, where μ is called the friction coefficient. If the contact is sliding, then $f_t = \mu f_n$, and the friction force opposes the direction of motion. The friction force is independent of the speed of sliding.

Often two friction coefficients are defined, a static friction coefficient μ_s and a kinetic (or sliding) friction coefficient μ_k , where $\mu_s \geq \mu_k$. This implies that a larger friction force is available to resist initial motion, but once motion has begun, the resisting force decreases. Many other friction models have been developed with

different functional dependencies on factors such as the speed of sliding and the duration of static contact before sliding. All of these are aggregate models of complex microscopic behavior. For simplicity, we will assume the simplest Coulomb friction model with a single friction coefficient μ . This model is reasonable for hard, dry materials. The friction coefficient depends on the two materials in contact, and typically ranges from 0.1 to 1.

Figure 27.3a shows that this friction law can be interpreted in terms of a *friction cone*. The set of all forces that can be applied to the disk by the supporting line is constrained to be inside this cone. Correspondingly, any force the disk applies to the support is inside the negative of the cone. The half-angle of the cone is $\beta = \tan^{-1} \mu$, as shown in Fig. 27.4. If the disk slips to the left on the support, the force the support applies to it acts on the right edge of the friction cone, with a magnitude determined by the normal force.

If we choose a coordinate frame, the force \mathbf{f} applied to the disk by the support can be expressed as a wrench

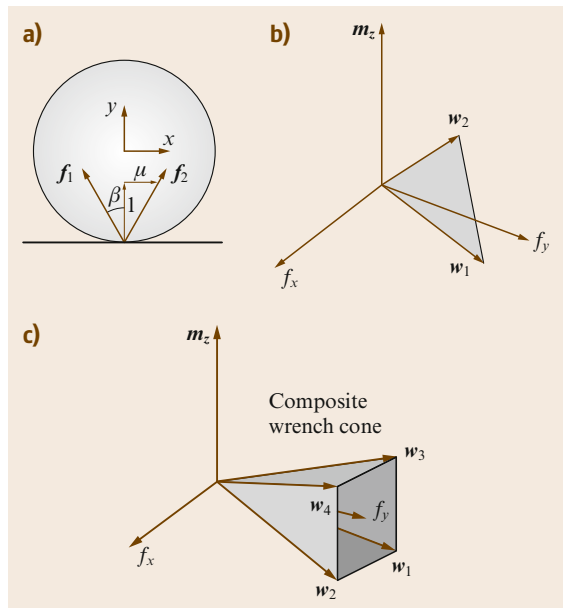


Fig. 27.3 (a) A planar friction cone. (b) The corresponding wrench cone. (c) An example composite wrench cone resulting from two frictional contacts

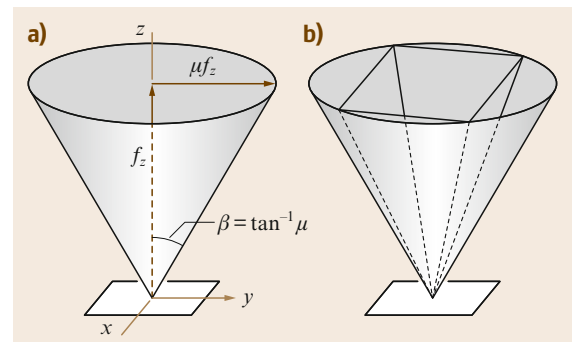


Fig. 27.4 (a) A spatial friction cone. The half-angle of the cone is $\beta = \tan^{-1} \mu$. (b) An inscribed pyramidal approximation to the friction cone. A more accurate inscribed pyramidal approximation can be used by increasing the number of faces of the pyramid. Depending on the application, a circumscribed pyramid could be used instead of an inscribed pyramid

$\mathbf{w} = [(\mathbf{x} \times \mathbf{f})^\top, \mathbf{f}^\top]^\top$, where \mathbf{x} is the contact location. Thus the friction cone turns into a wrench cone, as shown in Fig. 27.3b. The two edges of the planar friction cone give two half-lines in the wrench space, and the wrenches that can be transmitted to the part through the contact are all nonnegative linear combinations of basis vectors along these edges. If \mathbf{w}_1 and \mathbf{w}_2 are basis vectors for these wrench cone edges, we write the wrench cone as

$$\mathcal{WC} = \{k_1 \mathbf{w}_1 + k_2 \mathbf{w}_2 \mid k_1, k_2 \geq 0\}.$$

If there are multiple contacts acting on a part, then the total set of wrenches that can be transmitted to the part through the contacts is the nonnegative linear combination of all the individual wrench cones \mathcal{WC}_i ,

$$\begin{aligned} \mathcal{WC} &= \text{pos}(\{\mathcal{WC}_i\}) \\ &= \left\{ \sum_i k_i \mathbf{w}_i \mid \mathbf{w}_i \in \mathcal{WC}_i, k_i \geq 0 \right\}. \end{aligned}$$

This composite wrench cone is a convex polyhedral cone rooted at the origin. An example composite wrench cone arising from two planar frictional contacts is shown in Fig. 27.3c. If the composite wrench cone is the entire wrench space, then the contacts can provide a force closure grasp (Chap. 28).

In the spatial case, the friction cone is a circular cone, defined by

$$\sqrt{f_x^2 + f_y^2} \leq \mu f_z, f_z \geq 0 \quad (27.9)$$

when the contact normal is in the $+z$ -direction (Fig. 27.4). The resulting wrench cone, and composite wrench cone $\text{pos}(\{\mathcal{WC}_i\})$ for multiple contacts, is a convex cone rooted at the origin, but it is not polyhedral. For computational purposes, it is common to approximate circular friction cones as pyramidal cones, as shown in Fig. 27.4. Then individual and composite wrench cones become polyhedral convex cones in the six-dimensional wrench space.

If a contact or set of contacts acting on a part is ideally force controlled, the wrench \mathbf{w}_{ext} specified by the controller must lie within the composite wrench cone corresponding to those contacts. Because these force-controlled contacts choose a subset of wrenches (possibly a single wrench) from this wrench cone, the total composite wrench cone that can act on the part (including other non-force-controlled contacts) may no longer be a homogeneous cone rooted at the origin. This is roughly analogous to the case of velocity-controlled contacts in Sect. 27.2.1, which results in a feasible twist set for the part that is not a cone rooted at the origin. Ideal robot manipulators may be controlled by position

control, force control, hybrid position–force control, or some other scheme. The control method must be compatible with the parts' contacts with each other and the environment to prevent excessive forces [27.33].

27.3.1 Graphical Planar Methods

Just as homogeneous twist cones for planar problems can be represented as convex signed (+ or –) CoR regions in the plane, homogeneous wrench cones for planar problems can be represented as convex signed regions in the plane. This is called *moment labeling* [27.14, 34]. Given a collection of lines of force in the plane (e.g., the edges of friction cones from a set of point contacts), the set of all nonnegative linear combinations of these can be represented by labeling all the points in the plane with either a '+' if all resultants make nonnegative moment about that point, a '–' if all make nonpositive moment about that point, a '±' if all make zero moment about that point, and a blank label if there exist resultants making positive moment and resultants making negative moment about that point.

The idea is best illustrated by an example. In Fig. 27.5a, a single line of force is represented by labeling the points to the left of the line with a + and points to the right of the line with a –. Points on the line are labeled ±. In Fig. 27.5b, another line of force is added. Only the points in the plane that are consistently labeled for both lines of force retain their labels; inconsistently labeled points lose their labels. Finally, a third line of force is added in Fig. 27.5c. The result is a single region labeled +. A nonnegative combination of the three lines of force can create any line of force in the plane that passes around this region in a counterclockwise sense. This representation is equivalent to a homogeneous convex wrench cone representation.

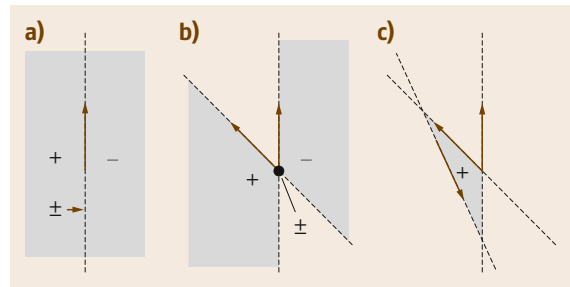


Fig. 27.5 (a) Representing a line of force by moment labels. (b) Representing the nonnegative linear combinations of two lines of force by moment labels. (c) Nonnegative linear combinations of three lines of force

27.3.2 Duality of Contact Wrenches and Twist Freedoms

Our discussion of kinematic constraints and friction should make it apparent that, for any point contact and contact label, the number of equality constraints on the part's motion caused by that contact is equal to the number of wrench freedoms it provides. For example, a breaking contact \mathfrak{b} provides zero equality constraints on the part motion and also allows no contact force. A fixed contact \mathfrak{f} provides three motion constraints (the motion of a point on the part is specified) and three freedoms on the contact force: any wrench in the interior of the contact wrench cone is consistent with the contact mode. Finally, a slipping contact \mathfrak{s} provides one equality motion constraint (one equation on the part's motion must be satisfied to maintain the contact), and for a given motion satisfying the constraint, the contact wrench has only one freedom, the magnitude of the contact wrench on the edge of the friction cone and opposite the slipping direction. In the planar case, the motion constraints and wrench freedoms for \mathfrak{b} , \mathfrak{s} , and \mathfrak{f} contacts are zero, one, and two, respectively.

At each contact, the force and velocity constraints can be represented by an $n \times 6$ *constraint* or *selection* matrix, \mathbf{H} [27.3], at the contact interface. This constraint matrix works like a filter to transmit or deny certain components of motion across the contact interface. By the same token, forces/moments applied across the contact interface are also filtered by the same constraint matrix, \mathbf{H}^\top , in a dual relationship. Three typical contact models are illustrated below. Contact forces and wrenches are measured in a frame at the contact, with the contact normal in the $+z$ -direction.

- The *point contact without friction* model. Only a normal force f_z can be exerted between the contacting bodies:

$$f_z \geq 0, \quad \mathbf{w} = \mathbf{H}^\top f_z = (0 \ 0 \ 1 \ 0 \ 0 \ 0)^\top f_z. \quad (27.10)$$

- The *point contact with friction* model includes tangential friction forces, f_x and f_y , in addition to normal force f_z ,

$$f_z \geq 0; |f_t| = \sqrt{f_x^2 + f_y^2} \leq \mu f_z, \quad \mathbf{w} = \mathbf{H}^\top \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ f_z \end{pmatrix}, \quad (27.11)$$

where f_x and f_y are the x - and y -components of the tangential contact force, and f_z is the normal force.

- The *soft finger* contact model with a finite contact patch allows, in addition to the friction and normal forces, a torsional moment with respect to the contact normal [27.35–39]. We have

$$f_z \geq 0; \quad \mathbf{w} = \mathbf{H}^\top \begin{pmatrix} f_x \\ f_y \\ f_z \\ m_z \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \\ f_z \\ m_z \end{pmatrix}, \quad (27.12)$$

where m_z is the moment with respect to the normal of contact. The finite contact area assumed by the soft contact interface results in the application of a friction moment in addition to the traction forces. If the resultant force on the tangential plane of contact is denoted as $f_t = \sqrt{f_x^2 + f_y^2}$ and the moment with respect to the contact normal is m_z , the following elliptical equation represents the relationship between the force and moment at the onset of sliding:

$$\frac{f_t^2}{a^2} + \frac{m_z^2}{b^2} = 1, \quad (27.13)$$

where $a = \mu f_z$ is the maximum friction force, and $b = (m_z)_{\max}$ is the maximum moment defined in equation (27.32). Further readings on this subject can be found in [27.35–42].

As can be appreciated from the three cases above, the rows of \mathbf{H} are the directions along which contact forces are supported. Conversely, relative motions of the two objects are constrained along these same directions: $\mathbf{H}\dot{\mathbf{q}} = 0$. Thus, the constraint matrix works like a kinematic filter which dictates the components of motions that can be transmitted through the contact interface. The introduction of this matrix makes it easy to model the contact mechanics in analysis using the dual kinematic relationship between force/moment and motion, as described in Table 27.1. For example, the constraint filter matrix in (27.12) describes that all three components of forces and one component of the moment with respect to the normal of contact can be transmitted through the contact interface of a soft finger.

In Chap. 28, when multiple contacts are considered, such \mathbf{H} matrices associated with each finger can be con-

Table 27.1 Duality of the kinematics of contact interface. The force/motion relationships for finger and grasped object are dual to each other in the joint space, contact interface, and the Cartesian space of the object. In this table, the notation of θ , \mathbf{x}_f , \mathbf{x}_p , and \mathbf{x}_b are the displacement in the joint space, the Cartesian space at the contact point on the finger, the Cartesian space at the contact point on the object, and the reference frame on the object, respectively, with \mathbf{x}_{tr} denoting the transmitted components. The δ prefix denotes infinitesimal changes in the named coordinates. The forces with corresponding subscripts are referenced to the corresponding coordinates as those of the displacement described above

Motion	$\mathbf{J}_\theta \quad \delta\theta = \delta\mathbf{x}_f$ ($6 \times m$)($m \times 1$) (6×1)	\rightleftharpoons	$\mathbf{H} \quad \delta\mathbf{x}_f = \delta\mathbf{x}_{tr} = \mathbf{H} \quad \delta\mathbf{x}_p$ ($n \times 6$)(6×1) ($n \times 1$) ($n \times 6$)(6×1)	\rightleftharpoons	$\mathbf{J}_c \quad \delta\mathbf{x}_b = \delta\mathbf{x}_p$ (6×6)(6×1) (6×1)
Force	Joints $\mathbf{J}_\theta^T \quad \mathbf{f}_f = \boldsymbol{\tau}$ ($m \times 6$) (6×1) ($m \times 1$)	\rightleftharpoons	Contact $\mathbf{f}_f = \mathbf{H}^T \quad \mathbf{f}_{tr} = \mathbf{f}_p$ (6×1) ($6 \times n$) ($n \times 1$) (6×1)	\rightleftharpoons	Object $\mathbf{J}_c^T \quad \mathbf{f}_p = \mathbf{f}_b$ (6×6) (6×1) (6×1)

catenated in an augmented matrix to work in conjunction with the grasp and Jacobian matrices for the analysis of grasping and manipulation.

Table 27.1 summarizes the dual relationship between the force/moment and displacement – a result that is derived from the principle of virtual work. In Table 27.1, \mathbf{J}_θ

is the joint Jacobian matrix relating joint velocities to fingertip velocities, and \mathbf{J}_c is the Cartesian coordinate transformation matrix relating the contact point with the centroidal coordinates of the grasped object. In addition, the number of degrees of freedom (DOFs) in the task space is n and the number of DOFs in the joint spaces is m .

27.4 Rigid-Body Mechanics with Friction

The manipulation planning problem is to choose the motions or forces applied by manipulator contacts so that the part (or parts) moves as desired. This requires solving the subproblem of determining the motion of parts given a particular manipulator action.

Let $\mathbf{q} \in \mathbb{R}^n$ be local coordinates describing the combined configuration of the system consisting of one or more parts and robot manipulators, and let $\mathbf{w}_i \in \mathbb{R}^6$ represent a wrench at an active contact i , measured in the common coordinate frame O . Let $\mathbf{w}_{all} \in \mathbb{R}^{6k}$ be the vector obtained by stacking the \mathbf{w}_i , $\mathbf{w}_{all} = (\mathbf{w}_1^T \mathbf{w}_2^T \dots \mathbf{w}_k^T)^T$ (where there are k contacts), and let $\mathbf{A}(\mathbf{q}) \in \mathbb{R}^{n \times 6k}$ be a matrix indicating how (or if) each contact wrench acts on each part. (Note that a contact wrench \mathbf{w}_i acting on one part means that there is a contact wrench $-\mathbf{w}_i$ acting on the other part in contact.) The problem is to find the contact forces \mathbf{w}_{all} and the system acceleration $\ddot{\mathbf{q}}$, given the state of the system $(\mathbf{q}, \dot{\mathbf{q}})$, the system mass matrix $\mathbf{M}(\mathbf{q})$ and resulting Coriolis matrix $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$, the gravitational forces $\mathbf{g}(\mathbf{q})$, the control forces $\boldsymbol{\tau}$, and the matrix $\mathbf{T}(\mathbf{q})$ indicating how the control forces $\boldsymbol{\tau}$ act on the system. (Alternatively, if we view the manipulators as position controlled, the manipulator components of $\ddot{\mathbf{q}}$ can be directly specified and the corresponding components of the control forces $\boldsymbol{\tau}$ solved for.) One way to solve this problem is to (a) enumerate the set of all possible

contact modes for the current active contacts, and (b) for each contact mode, determine if there are wrenches \mathbf{w}_{all} and accelerations $\ddot{\mathbf{q}}$ that satisfy the dynamics

$$\mathbf{A}(\mathbf{q})\mathbf{w}_{all} + \mathbf{T}(\mathbf{q})\boldsymbol{\tau} - \mathbf{g}(\mathbf{q}) = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} \quad (27.14)$$

and that are consistent with the contact mode's kinematic constraints (constraints on $\ddot{\mathbf{q}}$) and friction cone force constraints (constraints on \mathbf{w}_{all}). This formulation is quite general and applies to multiple parts in contact. Equation (27.14) may be simplified by appropriate representations of the configurations and velocities of rigid-body parts (for example, using angular velocities of the rigid bodies instead of derivatives of local angular coordinates).

This formulation leads to some surprising conclusions: there may be multiple solutions to a particular problem (ambiguity) or there may be no solutions (inconsistency) [27.7–14]. This strange behavior arises from the status of Coulomb's law as an approximate law, and it disappears for zero friction, or friction that is small enough. Despite this weakness of the Coulomb friction law, it is a useful approximation. Nonetheless, if we would like to prove (under the Coulomb model) that a particular desired part motion occurs, we generally must also show that no other motion can occur. Other-

wise we have only shown that the desired motion is one of the possible outcomes.

27.4.1 Complementarity

The conditions that each contact provides an equal number of motion constraints and wrench freedoms (Sect. 27.3.2) can be written as *complementarity* conditions. Thus the problem of solving (27.14), subject to contact constraints, can be formulated as a complementarity problem (CP) [27.12, 13, 43–46]. For planar problems, or spatial problems with approximate pyramidal friction cones, the problem is a linear complementarity problem [27.47]. For circular spatial friction cones, the problem is a nonlinear complementarity problem, due to the quadratic constraints describing the cones. In either case, standard algorithms can be used to solve for possible contact modes and part motions.

Alternatively, assuming linearized friction cones, we can formulate a linear constraint satisfaction program (LCSP) for each contact mode (e.g., a linear program with no objective function). The contact mode places linear constraints on the part accelerations, and the solver solves for the part accelerations and the nonnegative coefficients multiplying each of the edges of the friction cones, subject to the dynamics (27.14). Each LCSP with a feasible solution represents a feasible contact mode.

27.4.2 Quasistatic Assumption

A common assumption in robot manipulation planning is the *quasistatic* assumption. This assumption says that parts move slowly enough that inertial effects are negligible. This means that the right- and left-hand sides of (27.14) are zero. With this assumption, we often solve for part *velocities* rather than accelerations. These velocities must be consistent with the kinematic constraints and force constraints, and forces acting on the parts must always sum to zero.

27.4.3 Examples

While rigid-body mechanics problems with friction are usually solved using computational tools for CPs and LCSPs, equivalent graphical methods can be used for some planar problems to assist the intuition. As a simple example, consider the pipe clamp in Fig. 27.6 [27.14]. Under the external wrench w_{ext} , does the clamp slide down the pipe, or does it remain fixed in place? The figure uses moment labeling to represent the composite wrench cone of contact forces that can act on the

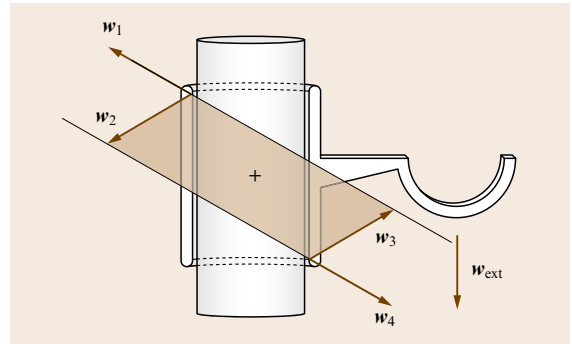


Fig. 27.6 The moment-labeling representation of the composite wrench cone of contact forces the pipe can apply to the pipe clamp. In this figure, the wrench w_{ext} applied to the pipe clamp can be resisted by forces within the composite wrench cone

clamp from the pipe. The wrench w_{ext} acting on the clamp can be exactly balanced by a wrench in the composite wrench cone. This is evident from the fact that the wrench opposing w_{ext} passes around the + labeled region in a counterclockwise sense, meaning that it is contained in the contact wrench cone. As a result, static equilibrium (the $f \neq f$ contact mode) is a feasible solution for the pipe clamp. To arrive at the same result using an LCSP, label the unit wrenches at the edges of the friction cones w_1, \dots, w_4 . Then the clamp can remain at rest if there exist coefficients a_1, \dots, a_4 such that

$$a_1, a_2, a_3, a_4 \geq 0, \\ a_1 w_1 + a_2 w_2 + a_3 w_3 + a_4 w_4 + w_{\text{ext}} = 0.$$

To show that static equilibrium is the only solution, all other contact modes must be ruled out. Note that, if the friction coefficient at the contacts is too small, the clamp will fall under w_{ext} .

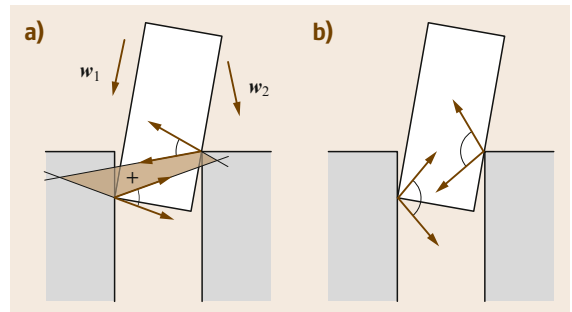


Fig. 27.7 (a) The peg will proceed into the hole under the external wrench w_1 but will get stuck under the wrench w_2 . (b) The peg is wedged

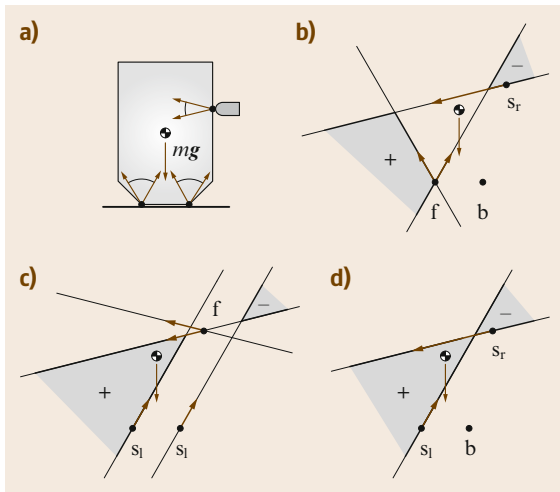


Fig. 27.8 (a) A finger moves *left* into a planar block resting on a table in gravity. The contact friction cones are shown. (b) Possible contact forces for the contact label f at the leftmost bottom contact, b at the rightmost bottom contact, and r at the pushing contact. This contact mode fbr corresponds to the block tipping over the leftmost contact. Note that the contact wrench cone (represented by moment labels) can provide a force that exactly balances the gravitational force. Thus this contact mode is quasistatically possible. (c) The contact wrench cone for the llf contact mode (block slides *left* on the table) cannot balance the gravitational force. This contact mode is not quasistatically possible. (d) The contact wrench cone for the lbr contact mode (block slides and tips) is not quasistatically possible

Analysis of the classic peg-in-hole problem is similar to that of the pipe clamp. Figure 27.7a shows an angled peg making two-point contact with the hole. If we apply the wrench w_1 , the forces from the two-point contact cannot resist, so the ff mode contact mode is not a possibility, and the peg will continue to move into the hole. If we apply the wrench w_2 , however, the contacts can resist, and the peg will get stuck. This is called *jamming* [27.48]. If there is higher friction at the contacts, as in Fig. 27.7b, then each friction cone may be able to see the base of the other, and the feasible contact wrenches span the entire wrench space [27.49]. In this case, any wrench we apply may be resisted by the contacts. The peg is said to be *wedged* [27.48]. Whether or not the peg will actually resist our applied wrench depends on how much internal force acts between the contacts. This cannot be answered by our rigid-body model; it requires a compliant contact model, as discussed in Sect. 27.6.

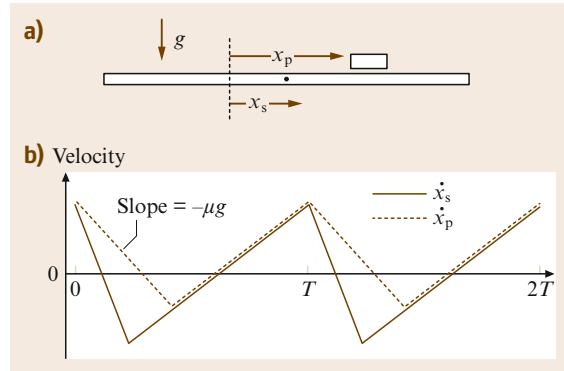


Fig. 27.9 (a) A part supported by a horizontally vibrating surface. (b) Friction between the part and the surface causes the part to always try to catch up to the surface, but its acceleration is bounded by $\pm\mu g$. The asymmetric motion of the surface gives the part an average positive velocity over a cycle

As a slightly more complex quasistatic example, consider a block on a table being pushed by a finger that moves horizontally to the left (Fig. 27.8) [27.50, 51]. Does the part begin to tip over, slide, or both tip and slide? The corresponding contact modes are fbs_r , s_1s_1f , and s_1lbs_r , as shown in Fig. 27.8. The figure also shows the composite contact wrench cones using moment labels. It is clear that quasistatic balance between the gravitational force and the contact wrench cone can only occur for the tipping without slipping contact mode fbs_r . Therefore, the only quasistatic solution is that the block begins to tip without sliding, and the speed of this motion is determined by the speed of the finger's motion. The graphical construction can be used to confirm our intuition that tipping of the block occurs if we push high on the block or if the friction coefficient at the block's support is high. Try it by pushing a can or glass. Quasistatically, the height of the center of mass is immaterial.

A final example is given in Fig. 27.9. A part of mass m and friction coefficient μ is supported by a horizontal surface that moves periodically in the horizontal direction. The periodic motion consists of a large negative acceleration for a short duration and a smaller positive acceleration for a longer duration. The horizontal friction force that can be applied to the part is bounded by $\pm\mu mg$, so the part's horizontal acceleration is bounded by $\pm\mu g$. As a result, the part cannot keep up with the surface as it accelerates backward. Instead, the part slips forward with respect to the surface, which means that the maximal friction force acts in the negative direction, and the part attempts to slow down to the surface's ve-

locity. Eventually the part catches up with the surface as it executes its slow forward acceleration, which is less than μg . Once the part catches up to the surface, it sticks to it, until the next backward acceleration phase. As is evident in Fig. 27.9, the average part velocity over a cycle is positive, so the part moves forward on the surface [27.52, 53]. This idea can be extended to cre-

ate a wide variety of frictional force fields on a rigid plate vibrated with three degrees of freedom in a horizontal plane [27.54, 55] or with a full six degrees of freedom [27.56].

For further examples of manipulation planning using the rigid-body model with Coulomb friction, see [27.14] and the references therein.

27.5 Pushing Manipulation

The friction limit surface (Sect. 27.7) is useful for analyzing pushing manipulation, as it describes the friction forces that can occur as a part slides over a support surface. When the part is pushed with a wrench contained within the limit surface, friction between the part and the support resists the pushing wrench and the part remains motionless. When the part slides quasistatically, the pushing wrench w lies on the limit surface, and the part's twist t is normal to the limit surface at w (Fig. 27.10). When the part translates without rotating, the friction force magnitude is μmg , where m is the part's mass and g is gravitational acceleration. The force applied by the part to the surface is directed through the part's center of mass in the direction of translation.

If the pushing wrench makes positive moment about the part's center of mass, the part will rotate counter-clockwise (CCW), and if it makes negative moment

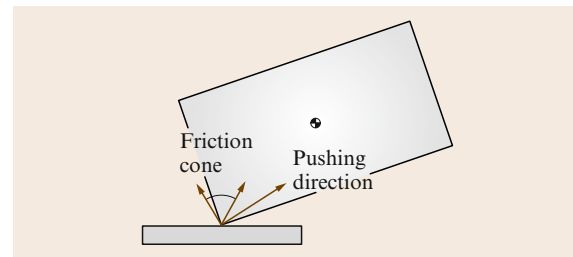


Fig. 27.11 The pusher's motion direction votes for counter-clockwise rotation of the part, but it is outvoted by the two edges of the friction cone, which both indicate clockwise rotation

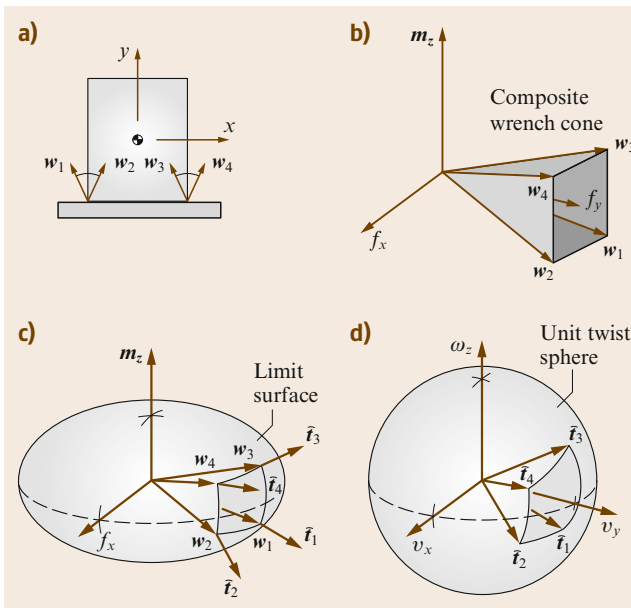


Fig. 27.10 (a) Contact between a pusher and a sliding object. (b) The composite contact wrench cone. (c) Mapping the wrench cone through the limit surface to twists of the object. (d) The unit twists that can result from forces in the composite wrench cone

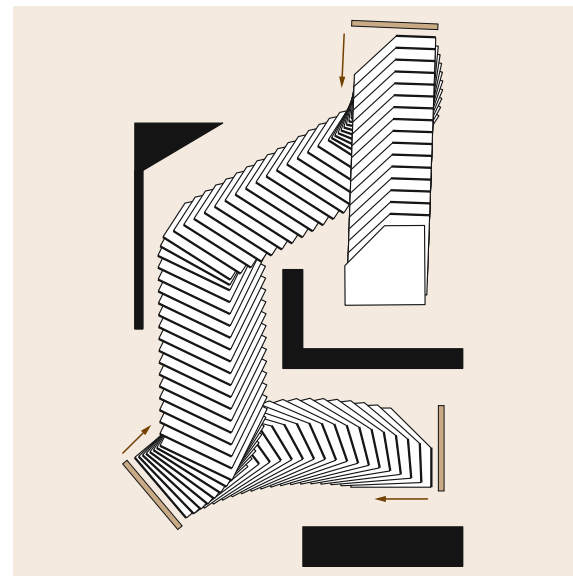


Fig. 27.12 Stable pushes can be used to maneuver a part among obstacles

about the part's center of mass, it will rotate clockwise (CW). Similarly, if the contact point on the part moves along a line that passes around the part's center of mass in a CW (respectively, CCW) sense, then the part will rotate CW (CCW). From these two observations and considering all possible contact modes, we can conclude that a part pushed at a point contact will rotate CW (CCW) if either (1) both edges of the contact friction cone pass CW (CCW) about the center of mass, or (2) one edge of the friction cone and the pushing direction at the contact both pass around the center of mass in a CW (CCW) sense [27.14, 57], (Fig. 27.11).

This observation allows pushing to be used to reduce uncertainty in part orientation. A series of pushes with a flat fence can be used to completely eliminate the uncertainty in the orientation of a polygonal

part [27.57–59]. Bounds on the rate of rotation of parts [27.14, 60, 61] allow the use of a sequence of stationary fences suspended above a conveyor belt to orient parts by pushing them as they are carried along by the conveyor [27.62, 63]. *Stable* pushing plans (Fig. 27.12) use pushing motions that are guaranteed to keep the part fixed to the pusher as it moves, even in the face of uncertainty in the part's pressure distribution [27.14, 64, 65]. Extensions of this work find pushing motions for planar assemblies of parts so that they remain fixed in their relative configurations during motion [27.66–68].

The examples above assume that pushing forces and support friction forces act in the same plane. Other work on pushing has considered three-dimensional effects, where pushing forces are applied above the support plane [27.69].

27.6 Contact Interfaces and Modeling

Contact interface is a general expression to describe the kinematics and kinetics of contacts. Contact applies in various contexts in robotics research; thus, it is more meaningful to refer to generic contact as an *interface*, which is not limited to fingers in grasping and manipulation. The concept of *contact interface* extends the traditional context of physical *contact*. It refers to an interface which imparts kinematic filtering as well as the duality of force/motion transmitted across the contact interface.

Consequently, a contact interface, whether it is rigid or deformable, can be considered as a *kinematic filter* which includes two characteristics: (a) motion and force transmission, and (b) kinematic duality between force/moment and motion. Different contact interfaces will be described in the following sections.

Previous sections of this chapter have assumed rigid bodies in contact. In reality, however, all contacts are accompanied by some deformation of the objects. Often this is by design, as in the case of compliant robot fingertips. When deformation is non-negligible, elastic contact models can be used.

27.6.1 Modeling of Contact Interface

Contact modeling depends on the nature of bodies in contact, including their material properties, applied force, contact deformation, and elastic properties. This section discusses different contact models.

Rigid-Body and Point-Contact Models

With the rigid-body assumption, as discussed in the previous sections of this chapter, two models are often used: (a) *point contact without friction* and (b) *point contact with friction*. In the former case, the contact can only apply force in the direction normal to the contact. In the latter case, a tangential friction force is applied in addition to the normal force. The simplest analytical model for point contact with friction is the Coulomb friction model as presented in (27.9).

Hertzian Contact Model

Elastic contact modeling was first studied and formulated more than a century ago by Hertz in 1882 [27.70] based on contact between two linear elastic materials with a normal force which results in very small contact deformation. This is commonly called the *Hertzian* contact and can be found in most mechanics textbooks, such as [27.71, 72]. Hertz made two important explicit assumptions in order for his contact model to be applicable:

1. objects of linear elastic materials in contact
2. small contact deformation compared to the dimension of object

Hertz also conducted experiments using a spherical glass lens against a planar glass plate to validate the contact theory.

Two relevant results of the Hertzian contact theory applied to robotic contact interfaces are summarized as follows. The first pertains to the radius of the contact area. Hertz [27.70] studied the growth of the contact area as a function of the applied normal force N based on the linear elastic model. Based on ten experimental trials, he concluded that the radius of contact is proportional to the normal force raised to the power of $1/3$, which is consistent with the analytical results he derived based on the linear elastic model. That is, the radius of contact, a , is related to the normal force, N , by

$$a \propto N^{1/3}. \quad (27.15)$$

The second result pertains to the pressure distribution over the assumed symmetric contact area – a second-order pressure distribution of the nature of an ellipse or circle. For a symmetric and circular contact area, the pressure distribution is

$$p(r) = \frac{N}{\pi a^2} \sqrt{1 - \left(\frac{r}{a}\right)^2}, \quad (27.16)$$

where N is the normal force, a is the radius of contact, and r is the distance from the center of contact with $0 \leq r \leq a$.

Soft Contact Model

A typical contact interface between a soft finger and contact surface is illustrated in Fig. 27.13. In typical robotic contact interfaces, the materials of the fingertips are not linear elastic. A model that extends linear to nonlinear elastic contact was presented in [27.38] with a power-law equation which subsumes the Hertzian contact theory,

$$a = c N^\gamma, \quad (27.17)$$

where $\gamma = n/(2n + 1)$ is the exponent of the normal force, n is the strain-hardening exponent, and c is a constant depending on the size and curvature of the fingertip

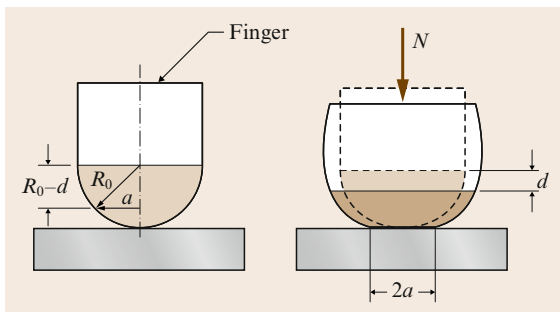


Fig. 27.13 An elastic soft fingertip with a hemispherical tip making contact with a rigid surface

as well as the material properties. Equation (27.17) is the new power law that relates the growth of the circular contact radius to the applied normal force for soft fingers. Note that the equation is derived assuming circular contact area. For linear elastic materials, the constant n is equal to 1 with $\gamma = 1/3$, resulting in the Hertzian contact model in (27.15). Therefore, the soft contact model in (27.17) subsumes the Hertzian contact model.

Other Models

Other models include anthropomorphic contacts and viscoelastic contacts. In modeling viscoelastic contacts, the creep or relaxation phenomenon is incorporated with a time-dependent function, in addition to the elastic deformation described in the general (27.17).

27.6.2 Pressure Distribution at Contacts

In Sect. 27.6.1, when the Hertzian contact theory is considered, the assumed pressure distribution for small elastic deformation is given in (27.16). As the radius of curvature of the two asperities increases and the material properties change to hyperelastic, the pressure distribution becomes more uniform [27.38, 73, 74]. Generalizing equation (27.16), the pressure distribution function for circular contact area with radius a is

$$p(r) = C_k \frac{N}{\pi a^2} \left[1 - \left(\frac{r}{a}\right)^k \right]^{1/k}, \quad (27.18)$$

where N is the normal force, a is the radius of contact, r is the radius with $0 \leq r \leq a$, k determines the shape of the pressure profile, and C_k is a coefficient that adjusts for the profile of pressure distribution over the contact area to satisfy the equilibrium condition. In (27.18), $p(r)$ is defined for $0 \leq r \leq a$. By symmetry, $p(r) = p(-r)$ when $-a \leq r \leq 0$, as shown in Fig. 27.14. When k becomes larger, the pressure distribution approaches uniform distribution, as shown in Fig. 27.14. It is also required that the integral of the pressure over the contact area be equal

Table 27.2 Values of the coefficient C_k in the pressure distribution (27.18),

k	Coefficient C_k
$k = 2$ (circular)	$C_2 = 1.5$
$k = 3$ (cubic)	$C_3 = 1.24$
$k = 4$ (quadruple)	$C_4 = 1.1441$
$k \rightarrow \infty$ (uniform)	$C_\infty = 1.0$

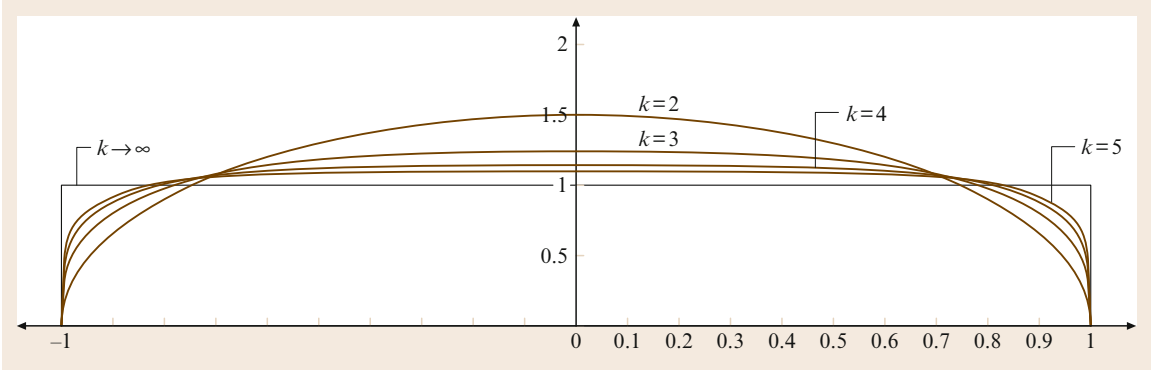


Fig. 27.14 Illustration of pressure distribution with respect to the normalized radius, r/a . The plot shows an axisymmetric pressure distribution with $k = 2, 3, 4, 5$, and ∞ . As k increases, the pressure profile becomes more uniform

to the normal force; that is,

$$\int_R p(r) dA = \int_{\theta=0}^{2\pi} \int_{r=0}^a p(r) r dr d\theta = N. \quad (27.19)$$

The coefficient C_k can be obtained by substituting (27.18) into (27.19). It is interesting to note that when (27.19) is integrated, both the normal force and radius of contact vanish, leaving only the constant C_k as follows

$$C_k = \frac{3}{2} \frac{k \Gamma\left(\frac{3}{k}\right)}{\Gamma\left(\frac{1}{k}\right) \Gamma\left(\frac{2}{k}\right)}, \quad (27.20)$$

where $k = 1, 2, 3, \dots$ are typically integer values (although noninteger k values are also possible) and $\Gamma()$ is the gamma function [27.75]. The numerical values of C_k for a few values of k are listed in Table 27.2 for reference. A normalized pressure distribution with respect to the normalized radius, r/a , is plotted in Fig. 27.14. As can be seen from the figure, when k approaches infinity, the pressure profile will become that of a uniformly distributed load with magnitude of $N/(\pi a^2)$, in which case $C_k = 1.0$.

For linear elastic materials, $k = 2$ can be used, although it is found that $k \cong 1.8$ is also appropriate in some cases [27.76]. For nonlinear elastic and viscoelastic materials, the value of k tends to be higher, depending on the properties of the material.

27.7 Friction Limit Surface

Section 27.3 introduced the notion of a wrench cone, describing the set of wrenches that can be applied through a point contact with friction. Any contact wrench is limited to lie within the surface of the cone. This gives rise to the general notion of a *friction limit surface* – the surface bounding the set of wrenches that can be applied through a given contact or set of contacts.

In this section we study the particular case of limit surfaces arising from planar contact patches [27.77, 78]. A planar contact patch occurs when a flat object slides on the floor, a soft robot finger presses against a face of a polyhedron, or a foot of a humanoid robot pushes on the ground. We would like to know what wrenches can be transmitted through such a contact. In the rest of this section, we focus on the properties of the limit sur-

face of a planar contact patch with a specified pressure distribution over the patch.

For ease of discussion, we will call one of the objects in contact the *part* (e.g., the flat object on the floor or the robot finger) and the other object a stationary *support*. We define a coordinate frame so that the planar contact patch is in the $z = 0$ plane, and let $p(\mathbf{r}) \geq 0$ be the contact pressure distribution between the part and support as a function of the location $\mathbf{r} = (x, y)^\top$. The friction coefficient at the contact patch is μ . If the planar velocity of the part is $\mathbf{t} = (\omega_z, v_x, v_y)^\top$, then the linear velocity at \mathbf{r} is

$$\mathbf{v}(\mathbf{r}) = (v_x - \omega_z y, v_y + \omega_z x)^\top,$$

and the unit velocity is $\hat{\mathbf{v}}(\mathbf{r}) = \mathbf{v}(\mathbf{r})/\|\mathbf{v}(\mathbf{r})\|$. The infinitesimal force applied by the part to the support at \mathbf{r} , in the

plane of sliding, is

$$d\mathbf{f}(\mathbf{r}) = [df_x(\mathbf{r}), df_y(\mathbf{r})]^\top = \mu p(\mathbf{r}) \hat{\mathbf{v}}(\mathbf{r}). \quad (27.21)$$

The total wrench the part applies to the support is

$$\begin{aligned} \mathbf{w} &= \begin{pmatrix} m_z \\ f_x \\ f_y \end{pmatrix} \\ &= \int_A \begin{pmatrix} x df_y(\mathbf{r}) - y df_x(\mathbf{r}) \\ df_x(\mathbf{r}) \\ df_y(\mathbf{r}) \end{pmatrix} dA, \end{aligned} \quad (27.22)$$

where A is the support area.

As expected, the wrench is independent of the speed of motion: for a given t_0 and all $\alpha > 0$, the wrench resulting from velocities αt_0 is identical. Viewing (27.22) as a mapping from twists to wrenches, we can map the surface of the unit twist sphere ($\|\hat{\mathbf{t}}\| = 1$) to a surface in the wrench space. This surface is the limit surface, and it is closed, convex, and encloses the origin of the wrench space (Fig. 27.15c). The portion of the wrench space enclosed by this surface is exactly the set of wrenches the part can transmit to the support. When the part slips on the support ($t \neq 0$), the contact wrench \mathbf{w} lies on the limit surface, and by the maximum-work inequality, the twist \mathbf{t} is normal to the limit surface at \mathbf{w} . If the pressure distribution $p(\mathbf{r})$ is finite everywhere, the limit surface is smooth and strictly convex, and the mapping from unit twists to unit wrenches, and vice versa, is continuous and one-to-one. The limit surface also satisfies the property $\mathbf{w}(-\hat{\mathbf{t}}) = -\mathbf{w}(\hat{\mathbf{t}})$.

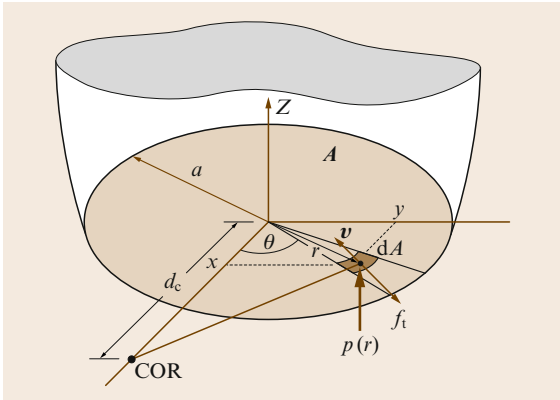


Fig. 27.15 Contact and coordinates for CoR and local infinitesimal area dA for numerical integration to construct the limit surface of soft fingers

27.7.1 The Friction Limit Surface at a Soft Contact Interface

Given a pressure distribution in (27.18) we can numerically construct the corresponding friction limit surface by using (27.21) and (27.22). For an infinitesimal contact area, the contact resembles a point contact. Hence, the Coulomb's law of friction can be employed.

In the following, we will exploit the kinematic property of center of rotation (CoR) with modeling symmetry to formulate equations for the magnitudes of friction force and moment on a circular contact patch. Figure 27.15 shows a circular contact patch with an instantaneous CoR. By moving (or scanning) the CoR along the x -axis, different possible combinations of friction force and moment can be obtained to construct the limit surface. More details can be found in [27.35–38, 42, 77–80].

The following derivation is for the total friction force (f_t) and moment (m_z) on the contact interface for the CoR at a distance d_c along the x -axis, as shown in Fig. 27.15. By varying the CoR distance d_c from $-\infty$ to ∞ , all possible combinations of (f_t, m_z) can be found in order to construct the entire friction limit surface.

The tangential force over the entire contact area can be obtained by integrating shear force on each infinitesimal areas, dA , on which the Coulomb's law of friction is observed, over the entire contact area, A . When Hertzian contact pressure distribution [27.70] is considered, use $k = 2$ in (27.18). Setting $\mathbf{r} = (x, y)^\top$ and $r = \|\mathbf{r}\|$, the total tangential force can be integrated using (27.21) to obtain

$$\mathbf{f}_t = \begin{pmatrix} f_x \\ f_y \end{pmatrix} = - \int_A \mu \hat{\mathbf{v}}(\mathbf{r}) p(r) dA, \quad (27.23)$$

where A denotes the circular contact region in Fig. 27.15, \mathbf{f}_t is the tangential force vector with the direction shown in Fig. 27.15, μ is the coefficient of friction, $\hat{\mathbf{v}}(\mathbf{r})$ is the unit vector in the direction of the velocity vector $\mathbf{v}(\mathbf{r})$ with respect to the CoR on the infinitesimal area dA at the location \mathbf{r} , and $p(r)$ is the pressure distribution at distance r from the center of contact. Since the pressure along the annular ring with distance r from the center is the same, we denote it by $p(r)$ instead of $p(\mathbf{r})$. The minus sign denotes the opposite directions of $\hat{\mathbf{v}}(\mathbf{r})$ and \mathbf{f}_t . Since we are primarily interested in the magnitude of the friction force and moment, we shall omit this sign in the later derivation when magnitudes are concerned.

Similarly, the moment about the z -axis, or the normal to the contact area, is

$$m_z = \int_A \mu \|\mathbf{r} \times \hat{\mathbf{v}}(\mathbf{r})\| p(r) dA, \quad (27.24)$$

where $\|\mathbf{r} \times \hat{\mathbf{v}}(\mathbf{r})\|$ is the magnitude of the cross product of the vectors \mathbf{r} and $\hat{\mathbf{v}}(\mathbf{r})$, whose direction is normal to the contact surface.

The unit vector $\hat{\mathbf{v}}(\mathbf{r})$ is related to the distance d_c from the origin to the CoR chosen on the x -axis from Fig. 27.15, and can be written as follows

$$\begin{aligned} \hat{\mathbf{v}}(\mathbf{r}) &= \frac{1}{\sqrt{(x-d_c)^2 + y^2}} \begin{bmatrix} -y \\ (x-d_c) \end{bmatrix} \\ &= \frac{1}{\sqrt{(r \cos \theta - d_c)^2 + (r \sin \theta)^2}} \\ &\quad \times \begin{bmatrix} -r \sin \theta \\ (r \cos \theta - d_c) \end{bmatrix}. \end{aligned} \quad (27.25)$$

Due to symmetry, $f_x = 0$ for all CoRs along the x -axis; therefore, the magnitude of the tangential force in the contact tangent plane is $f_t = f_y$. Substituting equations (27.18) and (27.25) into (27.23) and (27.24), we obtain

$$\begin{aligned} f_t &= \int_A \mu \frac{(r \cos \theta - d_c)}{\sqrt{r^2 + d_c^2 - 2rd_c \cos \theta}} C_k \\ &\quad \times \frac{N}{\pi a^2} \left[1 - \left(\frac{r}{a} \right)^k \right]^{\frac{1}{k}} dA. \end{aligned} \quad (27.26)$$

Similarly, the moment about the axis normal to the plane is

$$\begin{aligned} m_z &= \int_A \mu \frac{r^2 - r d_c \cos \theta}{\sqrt{r^2 + d_c^2 - 2rd_c \cos \theta}} C_k \\ &\quad \times \frac{N}{\pi a^2} \left[1 - \left(\frac{r}{a} \right)^k \right]^{\frac{1}{k}} dA. \end{aligned} \quad (27.27)$$

In (27.26) and (27.27), polar coordinates are defined such that $x = r \cos \theta$, $y = r \sin \theta$, and $dA = r dr d\theta$. We also introduce a normalized coordinate

$$\tilde{r} = \frac{r}{a}. \quad (27.28)$$

From (27.28), we can write $dr = a d\tilde{r}$. We also denote $\tilde{d}_c = d_c/a$, and assume that μ is constant throughout the contact area. Substituting the normalized coordinate into equation (27.26) and dividing both sides by μN , we

can derive

$$\begin{aligned} \frac{f_t}{\mu N} &= \frac{C_k}{\pi} \int_0^{2\pi} \int_0^1 \frac{(\tilde{r}^2 \cos \theta - \tilde{r} \tilde{d}_c)}{\sqrt{\tilde{r}^2 + \tilde{d}_c^2 - 2\tilde{r} \tilde{d}_c \cos \theta}} \\ &\quad \times (1 - \tilde{r}^k)^{\frac{1}{k}} d\tilde{r} d\theta. \end{aligned} \quad (27.29)$$

Substituting again $\tilde{r} = r/a$ and $dr = a d\tilde{r}$ into (27.27) and normalizing with $a\mu N$, we obtain

$$\begin{aligned} \frac{m_z}{a\mu N} &= \frac{C_k}{\pi} \int_0^{2\pi} \int_0^1 \frac{(\tilde{r}^3 \cos \theta - \tilde{r}^2 \tilde{d}_c)}{\sqrt{\tilde{r}^2 + \tilde{d}_c^2 - 2\tilde{r} \tilde{d}_c \cos \theta}} \\ &\quad \times (1 - \tilde{r}^k)^{\frac{1}{k}} d\tilde{r} d\theta. \end{aligned} \quad (27.30)$$

Equations (27.29) and (27.30) can be numerically integrated for a distance d_c or \tilde{d}_c to yield a point on the limit surface for a prescribed pressure distribution $p(r)$ given by (27.18). Both equations involve elliptic integrals whose closed-form solutions may not exist but can be evaluated numerically. When the CoR distance d_c varies from $-\infty$ to ∞ , all possible combinations of (f_t, m_z) can be obtained for plotting the friction limit surface.

27.7.2 Example of Constructing a Friction Limit Surface

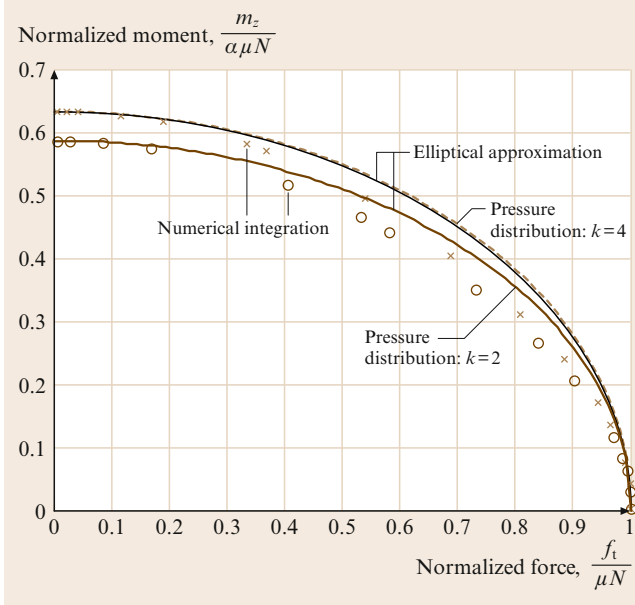
When the pressure distribution is fourth order with $k=4$, the coefficient from Table 27.2 is $C_4 = (6/\sqrt{\pi})(\Gamma(3/4)/\Gamma(1/4)) = 1.1441$. Equations (27.29) and (27.30) can be written as

$$\begin{aligned} \frac{f_t}{\mu N} &= 0.3642 \int_0^{2\pi} \int_0^1 \frac{(\tilde{r}^2 \cos \theta - \tilde{r} \tilde{d}_c)}{\sqrt{\tilde{r}^2 + \tilde{d}_c^2 - 2\tilde{r} \tilde{d}_c \cos \theta}} \\ &\quad \times (1 - \tilde{r}^4)^{\frac{1}{4}} d\tilde{r} d\theta, \\ \frac{m_z}{a\mu N} &= 0.3642 \int_0^{2\pi} \int_0^1 \frac{(\tilde{r}^3 \cos \theta - \tilde{r}^2 \tilde{d}_c)}{\sqrt{\tilde{r}^2 + \tilde{d}_c^2 - 2\tilde{r} \tilde{d}_c \cos \theta}} \\ &\quad \times (1 - \tilde{r}^4)^{\frac{1}{4}} d\tilde{r} d\theta. \end{aligned}$$

Numerical integration for different values of \tilde{d}_c yields pairs of $(f_t/(\mu N), m_z/(a\mu N))$. Plots of these pairs are shown in Fig. 27.16.

A reasonable approximation to these numerical results is given by the following equation of an ellipse

$$\left(\frac{f_t}{\mu N} \right)^2 + \left(\frac{m_z}{(m_z)_{\max}} \right)^2 = 1, \quad (27.31)$$



where the maximum moment $(m_z)_{\max}$ is

$$(m_z)_{\max} = \int_A \mu |r| C_k \frac{N}{\pi a^2} \left[1 - \left(\frac{r}{a} \right)^k \right]^{\frac{1}{k}} dA. \quad (27.32)$$

obtained by (27.27) with the CoR at $d_c = 0$. This defines the quarter-elliptical curves in Fig. 27.16. This approximation is the basis for constructing a three-

Fig. 27.16 Example limit surface obtained by numerical integration and elliptical approximation. The numerical integration is based on the pressure distribution, $p(r)$, and the coefficient, C_k . In this figure, the pressure distributions in (27.18) of both $k = 2$ and $k = 4$ are used

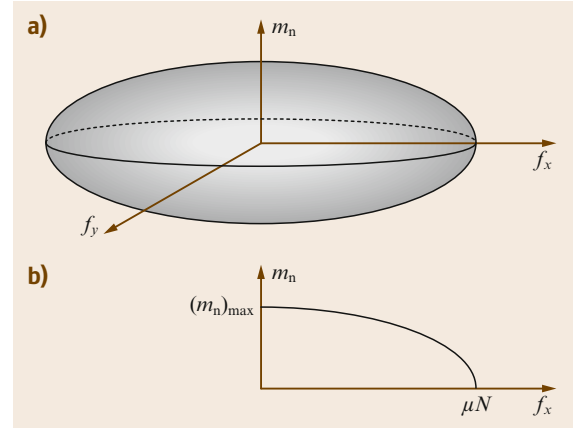


Fig. 27.17a,b Friction limit surface for soft fingers: (a) a 3-D ellipsoid representing the limit surface, and (b) a section of the ellipsoidal limit surface showing the coupled relationship between the force and moment

dimensional (3-D) ellipsoidal limit surface, as illustrated in Fig. 27.17, and is a good model for the soft contact described in Sect. 27.6.1. More details can be found in [27.35–37, 39, 40, 42, 78].

27.8 Contacts in Grasping and Fixture Designs

It is often important to relate the force and contact displacement (or deformation of contact interface) in grasping and fixture design in which deformable contacts are concerned. Furthermore, such a force–displacement relationship is typically nonlinear due to the nature of contact. A linear expression such as Hooke’s law cannot capture the instantaneous and overall characteristics of force and displacement for these contacts. In this section, we formulate and discuss such a relationship using the elastic contact model.

From Fig. 27.13 and the geometry of contact, (27.17), relating the contact radius with normal force, can be rewritten. Assuming $d \ll R_0$, the following equation can be derived [27.39]

$$N = c_d d^\zeta, \quad (27.33)$$

where c_d is a proportional constant, and ζ is

$$\zeta = \frac{1}{2\gamma}. \quad (27.34)$$

Both c_d and ζ can be obtained experimentally. The exponent ζ can also be obtained from γ through (27.34) if γ is already known for the fingertip. The range of the exponent in (27.33) is $\frac{3}{2} \leq \zeta < \infty$. In (27.33), the approach or the vertical depression of the fingertip, d , is proportional to the normal force raised to the power of 2γ (cf. (27.34)) which ranges from 0 to 2/3.

A plot of normal force versus displacement of contact is shown in Fig. 27.18 using (27.33) with a nonlinear power-law equation. Experimental data of contact between a soft finger and flat surface show consistent results when compared with Fig. 27.18 [27.39].

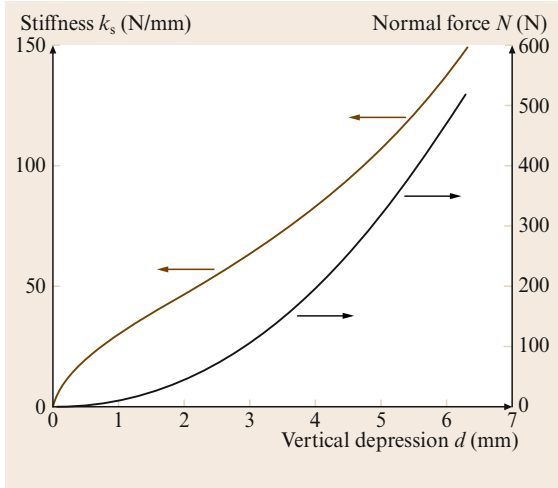


Fig. 27.18 The typical load–depression curve in (27.33) shown with a scale to the right, and the contact stiffness, k_s , as given by (27.35) with a scale to the left

27.8.1 Contact Stiffness of Soft Fingers

The nonlinear contact stiffness of a soft finger is defined as the ratio of the change in normal force with respect to the change in vertical depression at the contact. The contact stiffness of soft fingers can be obtained by differentiating (27.33) as follows

$$k_s = \frac{\partial N}{\partial d} = c_d \zeta d^{\zeta-1} = \zeta \left(\frac{N}{d} \right). \quad (27.35)$$

Substitute (27.33) with $d = \left(\frac{N}{c_d} \right)^{1/\zeta}$ into (27.35) to derive

$$k_s = c_d^{\frac{1}{\zeta}} \zeta N^{\frac{\zeta-1}{\zeta}} = c_d^{2\gamma} \zeta N^{1-2\gamma}. \quad (27.36)$$

Thus, the nonlinear contact stiffness of soft fingers, derived in a succinct form in (27.35), is the product of the exponent ζ and the ratio of the normal force versus the approach, N/d . A typical contact stiffness as a function of vertical depression is plotted in Fig. 27.18. The stiffness shown in the figure increases with the force. The expressions for the stiffness, k_s , as a function of various parameter(s) are summarized in the following Table 27.3.

Table 27.3 Summary of equations of contact stiffness

	$f(d)$	$f(N)$	$f(N, d)$
k_s	$c_d \zeta d^{\zeta-1}$	$c_d^{\frac{1}{\zeta}} \zeta N^{\frac{\zeta-1}{\zeta}}$	$\zeta \left(\frac{N}{d} \right)$

The change in stiffness can be obtained by differentiating (27.35) to derive the following equations:

$$\frac{\partial k_s}{\partial d} = c_d \zeta (\zeta - 1) d^{\zeta-2}, \quad (27.37)$$

$$\frac{\partial k_s}{\partial N} = \frac{\zeta - 1}{d}. \quad (27.38)$$

Equation (27.35) suggests that the stiffness of soft contacts always increases because the ratio N/d always increases, with constant ζ for prescribed fingertip material and range of normal force. This is consistent with the observation that the contact stiffness becomes larger (i. e., stiffer) with larger depression and force. Additionally, (27.38) suggests that the contact stiffness k_s always increases with the normal force because $\partial k_s / \partial N > 0$ owing to $\zeta \geq 1.5$. Moreover, the change in stiffness with respect to the normal load is inversely proportional to the vertical depression, d , as derived in (27.38). This result asserts that the rate of increase in the contact stiffness will gradually become less and less as the normal load and vertical depression increase.

A summary of the equations for soft fingers is presented in Table 27.4. The range of the exponent of the general power-law equation in (27.17) is $0 \leq \gamma \leq 1/3$. The Hertzian contact has an exponent of $\gamma = 1/3$; therefore, the Hertzian contact theory for linear elastic materials, such as steel or other metallic fingertips, under small deformation, generally follow the Hertzian contact theory fairly well. The power-law equation in (27.17) should be employed for fingertips made of softer materials, such as soft rubber or silicone or even viscoelastic fingertips.

27.8.2 Application of Soft Contact Theory to Fixture Design

The preceding analysis and results can be applied in fixturing design and other applications that involve contact with finite areas [27.6]. In fixture design with soft contacts (e.g., copper surfaces) under relatively large deformation and load, the power-law equations in Table 27.4 should be considered in place of the Hertzian contact equation. In these cases, the Hertzian contact model is no longer accurate and should be replaced. On the other hand, if linear elastic materials are used in fixture design with relatively small deformation ($d/R_0 \leq 5\%$), the exponent should be taken as $\gamma = 1/3$ when applying the contact theory. Furthermore, the exponent γ was found to be material dependent and not geometry dependent in general [27.38]. Once the value of γ is determined

Table 27.4 Summary of the contact mechanics equations for linear elastic (when $\gamma = 1/3$ or $\zeta = 3/2$) and nonlinear elastic soft fingers

Description	Equation for soft fingers	Parameters
Power law	$a = c N^\gamma$	$0 \leq \gamma \leq \frac{1}{3}$
Pressure distribution	$p(r) = p(0) \left[1 - \left(\frac{r}{a} \right)^k \right]^{\frac{1}{k}}$	Typically $k \geq 1.8$
Contact approach	$N = c_d d^\zeta$	$\frac{3}{2} \leq \zeta \leq \infty$
Contact stiffness	$k_s = \zeta \frac{N}{d}$	Nonlinear

for the material (for example, using a tensile testing machine with the experimental procedure in [27.38]), it can be employed for the analysis of fixturing design using the relevant equations presented herewith.

27.9 Conclusions and Further Reading

In this chapter, both rigid-body and elastic models of contact interfaces are discussed, including kinematic constraints and the duality between the contact wrenches and twists. Contact forces that may arise with Coulomb friction are described. Multiple-contact manipulation tasks with rigid-body and Coulomb friction are presented. The friction limit surface is introduced and utilized to analyze pushing problems. An example of constructing a friction limit surface for soft contact is presented based on the formulation of force/moment of soft contact interface. Applications of these contact models in fixture analysis and design are presented with the modeling of contact interface. Many references in the bibliography section of this chapter provide further reading on this subject.

Manipulation modes include grasping, pushing, rolling, batting, throwing, catching, casting, and other kinds of quasistatic and dynamic manipulation [27.81].

This chapter presents an overview of contact modeling interfaces, with a particular focus on their use in manipulation tasks, including graspless or *nonprehensile* manipulation modes such as pushing. Analyses of grasps and fixtures are also presented. *Mason's* textbook [27.14] expands on a number of themes in this chapter, including the theory of polyhedral convex cones, graphical methods for planar problems, and applications to manipulation planning. Fundamental material on polyhedral cones can be found in [27.82] and their application to representations of twists and wrenches in [27.10, 83–85]. Twists and wrenches are elements of classical *screw theory*, which is covered in the texts [27.86, 87] and from the point of view of robotics in the texts [27.30, 88, 89].

In addition, several relevant chapters in this handbook provide further reading for theory and applications, for example, Chaps. 1 and 28.

References

27.1

A. Bicchi: On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation, *Int. J. Robot. Auton. Syst.* **13**, 127–147 (1994)

27.2

K. Harada, M. Kaneko, T. Tsuji: Rolling Based Manipulation for Multiple Objects, *Proceedings of IEEE Int. Conf. on Robotics and Automation* (San Francisco 2000) pp. 3888–3895

27.3

M.R. Cutkosky, I. Kao: Computing and Controlling The Compliance of a Robotic Hand, *IEEE Trans. Robot. Autom.*, **5**(2), 151–165 (1989)

27.4

M.R. Cutkosky, S.-H. Lee: Fixture Planning with Friction for Concurrent Product/Process Design, *NSF Process Planning* (1989)

27.5

S.-H. Lee, M. Cutkosky: Fixture planning with friction, *ASME J. Eng. Ind.* **113**(3), 320–327 (1991)

27.6

Q. Lin, J.W. Burdick, E. Rimon: A stiffness-based quality measure for compliant grasps and fixtures, *IEEE Trans. Robot. Autom.* **16**(6), 675–688 (2000), ,

27.7

P. Lötstedt: Coulomb friction in two-dimensional rigid body systems, *Z. Angew. Math. Mech.* **61**, 605–615 (1981)

- 27.8 P. Lötstedt: Mechanical systems of rigid bodies subject to unilateral constraints, *SIAM J. Appl. Math.* **42**(2), 281–296 (1982)
- 27.9 P.E. Dupont: The effect of Coulomb friction on the existence and uniqueness of the forward dynamics problem, *International Conference on Robotics and Automation (Nice 1992)* pp. 1442–1447
- 27.10 M.A. Erdmann: On a representation of friction in configuration space, *Int. J. Robot. Res.* **13**(3), 240–271 (1994)
- 27.11 K.M. Lynch, M.T. Mason: Pulling by pushing, slip with infinite friction, and perfectly rough surfaces, *Int. J. Robot. Res.* **14**(2), 174–183 (1995)
- 27.12 J.S. Pang, J.C. Trinkle: Complementarity formulations and existence of solutions of dynamic multi-rigid-body contact problems with Coulomb friction, *Math. Prog.* **73**, 199–226 (1996)
- 27.13 J.C. Trinkle, J.S. Pang, S. Sudarsky, G. Lo: On dynamic multi-rigid-body contact problems with Coulomb friction, *Z. Angew. Math. Mech.* **77**(4), 267–279 (1997)
- 27.14 M.T. Mason: *Mechanics of Robotic Manipulation* (MIT Press, Cambridge 2001)
- 27.15 Y.-T. Wang, V. Kumar, J. Abel: Dynamics of Rigid Bodies Undergoing Multiple Frictional Contacts, *Proceedings of IEEE Int. Conf. on Robotics and Automation (Nice, France 1992)* pp. 2764–2769
- 27.16 T.H. Speeter: Three-dimensional finite element analysis of elastic continua for tactile sensing, *Int. J. Robot. Res.* **11**(1), 1–19 (1992)
- 27.17 K. Dandekar, A.K. Srinivasan: A 3-Dimensional Finite Element Model of the Monkey Fingertip for Predicting Responses of Slowly Adapting Mechanoreceptors, *ASME Bioengineering Conference*, Vol. 29 (1995) pp. 257–258
- 27.18 N. Xydas, M. Bhagavat, I. Kao: Study of Soft-Finger Contact Mechanics Using Finite Element Analysis and Experiments, *Proc. IEEE Int. Conf. on Robotics and Automation*, ICRA (San Francisco, California 2000)
- 27.19 K. Komvopoulos, D.-H. Choi: Elastic finite element analysis of multi-asperity contacts, *J. Tribol.* **114**, 823–831 (1992)
- 27.20 L.T. Tenek, J. Argyris: *Finite Element Analysis for Composite Structures* (Kluwer Academic, Bosten 1998)
- 27.21 Y. Nakamura: Contact Stability Measure and Optimal Finger Force Control of Multi-Fingered Robot Hands, *Crossing Bridges: Advances in Flexible Automation and Robotics – The Proceedings of the USA-Japan Symposium on Flexible Automation (ASME, 1988)* pp. 523–528
- 27.22 Y.C. Park, G.P. Starr: Optimal Grasping Using a Multifingered Robot Hand, *Proceedings of the 1990 IEEE International Conference on Robotics and Automation (IEEE, Cincinnati, Ohio 1990)* pp. 689–694
- 27.23 E. Rimon, J. Burdick: On force and form closure for multiple finger grasps, *IEEE International Conference on Robotics and Automation (1996)* pp. 1795–1800
- 27.24 E. Rimon, J.W. Burdick: New bounds on the number of frictionless fingers required to immobilize planar objects, *J. Robot. Sys.* **12**(6), 433–451 (1995)
- 27.25 E. Rimon, J.W. Burdick: Mobility of bodies in contact—part I: A 2nd-order mobility index for multiple-finger grasps, *IEEE Trans. Robot. Autom.* **14**(5), 696–708 (1998)
- 27.26 D.J. Montana: The kinematics of contact and grasp, *Int. J. Robot. Res.* **7**(3), 17–32 (1988)
- 27.27 C.S. Cai, B. Roth: On the planar motion of rigid bodies with point contact, *Mechanism Machine Theory* **21**(6), 453–466 (1986)
- 27.28 C. Cai, B. Roth: On the spatial motion of a rigid body with point contact, *IEEE International Conference on Robotics and Automation (1987)* pp. 686–695
- 27.29 A.B.A. Cole, J.E. Hauser, S.S. Sastry: Kinematics and control of multifingered hands with rolling contact, *IEEE Trans. Autom. Control* **34**(4), 398–404 (1989)
- 27.30 R.M. Murray, Z. Li, S.S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1994)
- 27.31 F. Reuleaux: *The Kinematics of Machinery* (Dover, New York 1963), , reprint of MacMillan, 1876
- 27.32 C.A. Coulomb: Theorie des machines simples en ayant egard au frottement de leurs parties et a la roideur des cordages. In: *Memoires des mathematique et de physique presentes a l'Academie des Sciences* (Bachelier, Paris 1821)
- 27.33 Y. Maeda, T. Arai: Planning of graspless manipulation by a multifingered robot hand, *Adv. Robot.* **19**(5), 501–521 (2005)
- 27.34 M.T. Mason: Two graphical methods for planar contact problems, *IEEE/RSJ International Conference on Intelligent Robots and Systems (Osaka, Japan November 1991)* pp. 443–448
- 27.35 R. Howe, I. Kao, M. Cutkosky: Sliding of Robot Fingers Under Combined Torsion and Shear Loading, *Proceedings of 1988 IEEE International Conference on Robotics and Automation*, Vol. 1 (Philadelphia, Pennsylvania 1988) pp. 103–105
- 27.36 I. Kao, M.R. Cutkosky: Dextrous manipulation with compliance and sliding, *Int. J. Robot. Res.* **11**(1), 20–40 (1992)
- 27.37 R.D. Howe, M.R. Cutkosky: Practical force-motion models for sliding manipulation, *Int. J. Robot. Res.* **15**(6), 555–572 (1996)
- 27.38 N. Xydas, I. Kao: Modeling of contact mechanics and friction limit surface for soft fingers with experimental results, *Int. J. Robot. Res.* **18**(9), 941–950 (1999)
- 27.39 I. Kao, F. Yang: Stiffness and contact mechanics for soft fingers in grasping and manipulation, *IEEE Trans. Robot. Autom.* **20**(1), 132–135 (2004), ,
- 27.40 J. Jameson, L. Leifer: Quasi-Static Analysis: A Method for Predicting Grasp Stability, *Proceedings*

- of 1986 IEEE International Conference on Robotics and Automation (1986) pp. 876–883
- 27.41 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction: Part 2. Dynamics of motion, *Wear* **143**, 331–352 (1991)
- 27.42 P. Tiezzi, I. Kao: Modeling of viscoelastic contacts and evolution of limit surface for robotic contact interface, *IEEE Trans. Robot.* **23**(2), 206–217 (2007)
- 27.43 M. Anitescu, F. Potra: Formulating multi-rigid-body contact problems with friction as solvable linear complementarity problems, *ASME J. Nonlin. Dyn.* **14**, 231–247 (1997)
- 27.44 S. Berard, J. Trinkle, B. Nguyen, B. Roghani, J. Fink, V. Kumar: daVinci code: A multi-model simulation and analysis tool for multi-body systems, *IEEE International Conference on Robotics and Automation* (2007)
- 27.45 P. Song, J.-S. Pang, V. Kumar: A semi-implicit time-stepping model for frictional compliant contact problems, *Int. J. Numer. Methods Eng.* **60**(13), 2231–2261 (2004)
- 27.46 D. Stewart, J. Trinkle: An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and Coulomb friction, *Int. J. Numer. Methods Eng.* **39**, 2673–2691 (1996)
- 27.47 R.W. Cottle, J.-S. Pang, R.E. Stone: *The Linear Complementarity Problem* (Academic, New York 1992)
- 27.48 S.N. Simunovic: Force information in assembly processes, *International Symposium on Industrial Robots* (1975)
- 27.49 V.-D. Nguyen: Constructing force-closure grasps, *Int. J. Robot. Res.* **7**(3), 3–16 (1988),
- 27.50 K.M. Lynch: Toppling manipulation, *IEEE International Conference on Robotics and Automation* (1999)
- 27.51 M.T. Zhang, K. Goldberg, G. Smith, R.-P. Berretty, M. Overmars: Pin design for part feeding, *Robotica* **19**(6), 695–702 (2001)
- 27.52 D. Reznik, J. Canny: The Coulomb pump: a novel parts feeding method using a horizontally-vibrating surface, *IEEE International Conference on Robotics and Automation* (1998) pp. 869–874
- 27.53 A.E. Quaid: A miniature mobile parts feeder: Operating principles and simulation results, *IEEE International Conference on Robotics and Automation* (1999) pp. 2221–2226
- 27.54 D. Reznik, J. Canny: A flat rigid plate is a universal planar manipulator, *IEEE International Conference on Robotics and Automation* (1998) pp. 1471–1477
- 27.55 D. Reznik, J. Canny: C'mon part, do the local motion!, *IEEE International Conference on Robotics and Automation* (2001) pp. 2235–2242
- 27.56 T. Vose, P. Umbanhowar, K.M. Lynch: Vibration-induced frictional force fields on a rigid plate, *IEEE International Conference on Robotics and Automation* (2007)
- 27.57 M.T. Mason: Mechanics and planning of manipulator pushing operations, *Int. J. Robot. Res.* **5**(3), 53–71 (1986)
- 27.58 K.Y. Goldberg: Orienting polygonal parts without sensors, *Algorithmica* **10**, 201–225 (1993)
- 27.59 R.C. Brost: Automatic grasp planning in the presence of uncertainty, *Int. J. Robot. Res.* **7**(1), 3–17 (1988)
- 27.60 J.C. Alexander, J.H. Maddocks: Bounds on the friction-dominated motion of a pushed object, *Int. J. Robot. Res.* **12**(3), 231–248 (1993)
- 27.61 M.A. Peshkin, A.C. Sanderson: The motion of a pushed, sliding workpiece, *IEEE J. Robot. Autom.* **4**(6), 569–598 (1988)
- 27.62 M.A. Peshkin, A.C. Sanderson: Planning robotic manipulation strategies for workpieces that slide, *IEEE J. Robot. Autom.* **4**(5), 524–531 (1988)
- 27.63 M. Brokowski, M. Peshkin, K. Goldberg: Curved fences for part alignment, *IEEE International Conference on Robotics and Automation* (Atlanta 1993) pp. 467–473
- 27.64 K.M. Lynch: The mechanics of fine manipulation by pushing, *IEEE International Conference on Robotics and Automation* (Nice 1992) pp. 2269–2276
- 27.65 K.M. Lynch, M.T. Mason: Stable pushing: Mechanics, controllability, and planning, *Int. J. Robot. Res.* **15**(6), 533–556 (1996)
- 27.66 K. Harada, J. Nishiyama, Y. Murakami, M. Kaneko: Pushing multiple objects using equivalent friction center, *IEEE International Conference on Robotics and Automation* (2002) pp. 2485–2491
- 27.67 J.D. Bernheisel, K.M. Lynch: Stable transport of assemblies: Pushing stacked parts, *IEEE Trans. Autom. Sci. Eng.* **1**(2), 163–168 (2004)
- 27.68 J.D. Bernheisel, K.M. Lynch: Stable transport of assemblies by pushing, *IEEE Trans. Robot.* **22**(4), 740–750 (2006)
- 27.69 H. Mayeda, Y. Wakatsuki: Strategies for pushing a 3D block along a wall, *IEEE/RSJ International Conference on Intelligent Robots and Systems* (Osaka, Japan 1991) pp. 461–466
- 27.70 H. Hertz: On the Contact of Rigid Elastic Solids and on Hardness. In: *6: Assorted Papers by H. Hertz*, ed. by H. Hertz (MacMillan, New York 1882)
- 27.71 K.L. Johnson: *Contact Mechanics* (Cambridge Univ. Press, Cambridge 1985)
- 27.72 S.P. Timoshenko, J.N. Goodier: *Theory of Elasticity*, 3rd edn. (McGraw-Hill, New York 1970)
- 27.73 E. Wolf: *Progress in Optics* (North-Holland, Amsterdam 1992)
- 27.74 E.J. Nicolson, R.S. Fearing: The Reliability of Curvature Estimates from Linear Elastic Tactile Sensors, the Proceedings of the 1995 IEEE International Conference on Robotics and Automation (IEEE Press 1995)
- 27.75 M. Abramowitz, I. Stegun: *Handbook of Mathematical Functions with formulas, graphs, and*

- mathematical tables*, 7th edn. (Dover, New York 1972)
- 27.76 I. Kao, S.-F. Chen, Y. Li, G. Wang: Application of bio-engineering contact interface and MEMS in robotic and human augmented systems, *IEEE Robot. Autom. Mag.* **10**(1), 47–53 (2003)
- 27.77 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction. Part 1. Limit surface and moment function, *Wear* **143**, 307–330 (1991)
- 27.78 S. Goyal, A. Ruina, J. Papadopoulos: Planar sliding with dry friction. Part 2. Dynamics of motion, *Wear* **143**, 331–352 (1991)
- 27.79 J.W. Jameson: Analytic Techniques for Automated Grasp. Ph.D. Thesis (Department of Mechanical Engineering, Stanford University 1985)
- 27.80 S. Goyal, A. Ruina, J. Papadopoulos: Limit Surface and Moment Function Description of Planar Sliding, *Proceedings of 1989 IEEE International Conference on Robotics and Automation* (IEEE Computer Society, Scottsdale, Arizona 1989) pp. 794–799
- 27.81 K.M. Lynch, M.T. Mason: Dynamic nonprehensile manipulation: Controllability, planning, and experiments, *Int. J. Robot. Res.* **18**(1), 64–92 (1999)
- 27.82 A.J. Goldman, A.W. Tucker: Polyhedral convex cones. In: *Linear Inequalities and Related Systems*, ed. by H.W. Kuhn, A.W. Tucker (Princeton Univ. Press, Princeton 1956)
- 27.83 M.A. Erdman: A configuration space friction cone; IEEE/RSJ International Conference on Intelligent Robots and Systems (Osaka, 1991) pp. 455–460
- 27.84 M.A. Erdmann: Multiple-point contact with friction: Computing forces and motions in configuration space, IEEE/RSJ International Conference on Intelligent Robots and Systems (Yokohama, 1993) pp. 163–170
- 27.85 S. Hirai, H. Asada: Kinematics and statics of manipulation using the theory of polyhedral convex cones, *Int. J. Robot. Res.* **12**(5), 434–447 (1993)
- 27.86 R.S. Ball: *The Theory of Screws* (Cambridge Univ. Press, Cambridge 1900)
- 27.87 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Oxford Univ., Oxford 1978)
- 27.88 J.K. Davidson, K.H. Hunt: *Robots and Screw Theory* (Oxford Univ. Press, Oxford 2004)
- 27.89 J.M. Selig: *Geometric Fundamentals of Robotics*, 2nd edn. (Springer, Berlin Heidelberg 2005)

Grasping

28. Grasping

Domenico Prattichizzo, Jeffrey C. Trinkle

This chapter introduces fundamental models of grasp analysis. The overall model is a coupling of models that define contact behavior with widely used models of rigid-body kinematics and dynamics. The contact model essentially boils down to the selection of components of contact force and moment that are transmitted through each contact. Mathematical properties of the complete model naturally give rise to five primary grasp types whose physical interpretations provide insight for grasp and manipulation planning.

After introducing the basic model and types of grasps, this chapter focuses on the most important grasp characteristic: complete restraint. A grasp with complete restraint prevents loss of contact and thus is very secure. Two primary restraint properties are *form closure* and *force closure*. A form closure grasp guarantees maintenance of contact as long as the links of the hand and the object are well approximated as rigid and as long as the joint actuators are sufficiently strong. As will be seen, the primary difference between form closure and force closure grasps is the latter's reliance on contact friction. This translates into

28.1	Background	671
28.2	Models and Definitions	672
28.2.1	Velocity Kinematics	673
28.2.2	Dynamics and Equilibrium	676
28.3	Controllable Twists and Wrenches	677
28.3.1	Grasp Classifications	678
28.3.2	Limitations of Rigid-Body Assumption	679
28.3.3	Desirable Properties	679
28.4	Restraint Analysis	680
28.4.1	Form Closure	681
28.4.2	Force Closure	684
28.5	Examples	687
28.5.1	Example 1: Grasped Sphere	687
28.5.2	Example 2: Grasped Polygon in the Plane	691
28.5.3	Example 3: Hyperstatic Grasps	693
28.5.4	Example 4: Duality	696
28.5.5	Example 5: Form Closure	696
28.6	Conclusion and Further Reading	697
	References	698

requiring fewer contacts to achieve force closure than form closure.

28.1 Background

Mechanical hands were developed to give robots the ability to grasp objects of varying geometric and physical properties. The first robotic hand designed for dexterous manipulation was the Salisbury hand [28.1]. It has three three-jointed fingers; enough to control all six degrees of freedom of an object and the grip pressure. The fundamental grasp modeling and analysis done by Salisbury provides a basis for grasp synthesis and dexterous manipulation research which continues today. Some of the most mature analysis techniques are embedded in the software *GraspIt!* [28.2]. *GraspIt!*

contains models for several robot hands and provides tools for grasp selection, dynamic grasp simulation, and visualization.

The goal of this chapter is to give a thorough understanding of the all-important grasp properties of form and force closure. This will be done through detailed derivations of grasp models and discussions of illustrative examples. For an in-depth historical perspective and a treasure-trove bibliography of papers addressing a wide range of topics in grasping, the reader is referred to [28.3].

28.2 Models and Definitions

A mathematical model of grasping must be capable of predicting the behavior of the hand and object under the various loading conditions that may arise during grasping. Generally, the most desirable behavior is grasp maintenance in the face of unknown disturbing forces and moments applied to the object. Typically these disturbances arise from inertia forces which become appreciable during high-speed manipulation or applied forces such as those due to gravity. Grasp maintenance means that the contact forces applied by the hand are such that they prevent contact separation and unwanted contact sliding. The special class of grasps that can be maintained for every possible disturbing load is known as closure grasps. Figure 28.1 shows the Salisbury hand [28.1, 4] executing a closure grasp of an object by wrapping its fingers around it and pressing the object against its palm. Formal definitions, analysis, and computational tests for closure will be presented in Sect. 28.5.

Figure 28.2 illustrates some of the main quantities that will be used to model grasping systems. Assume that the links of the hand and the object are rigid and that there is a unique, well-defined tangent plane at each contact point. Let $\{N\}$ represent a conveniently chosen inertial frame fixed in the workspace. The frame $\{B\}$ is fixed to the object with its origin defined relative to $\{N\}$ by the vector $\mathbf{p} \in \mathbb{R}^3$, where \mathbb{R}^3 denotes three-dimensional Euclidean space. A convenient choice for \mathbf{p} is the center of mass of the object. The position of contact point i in $\{N\}$ is defined by the vector $\mathbf{c}_i \in \mathbb{R}^3$. At contact point i , we define a frame $\{C\}_i$, with axes $\{\hat{n}_i, \hat{t}_i, \hat{o}_i\}$ ($\{C\}_i$ is shown in *exploded view*). The unit vector \hat{n}_i contains \mathbf{c}_i is normal to the contact tangent

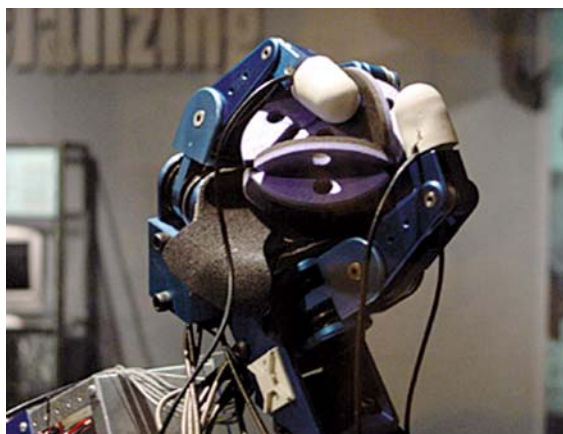


Fig. 28.1 The Salisbury hand grasping an object

plane, and is directed toward the object. The other two unit vectors are orthogonal and lie in the tangent plane of the contact.

Let the joints be numbered from 1 to n_q . Denote by $\mathbf{q} = [q_1 \cdots q_{n_q}]^T \in \mathbb{R}^{n_q}$ the vector of joint displacements, where the superscript T indicates matrix transposition. Also, let $\boldsymbol{\tau} = [\tau_1 \cdots \tau_{n_q}]^T \in \mathbb{R}^{n_q}$ represent joint loads (forces in prismatic joints and torques in revolute joints). These loads can result from actuator actions, other applied forces, and inertia forces. They could also arise from contacts between the object and hand. However, it will be convenient to separate joint loads into two components: those arising from contacts and those arising from all other sources. Throughout this chapter, noncontact loads will be denoted by $\boldsymbol{\tau}$.

Let $\mathbf{u} \in \mathbb{R}^{n_u}$ denote the vector describing the position and orientation of $\{B\}$ relative to $\{N\}$. For planar systems $n_u = 3$. For spatial systems, n_u is three plus the number of parameters used to represent orientation, typically three (for Euler angles) or four (for unit quaternions). Denote by $\mathbf{v} = [\mathbf{v}^T \boldsymbol{\omega}^T]^T \in \mathbb{R}^{n_v}$ the twist of the object described in $\{N\}$. It is composed of the translational velocity $\mathbf{v} \in \mathbb{R}^3$ of the point \mathbf{p} and the angular velocity $\boldsymbol{\omega} \in \mathbb{R}^3$ of the object, both expressed in $\{N\}$. A twist of a rigid body can be referred to any convenient frame fixed to the body. The components of the referred twist represent the velocity of the origin of the new frame and the angular velocity of the body, both expressed in the new frame. For a rigorous treatment of twists and

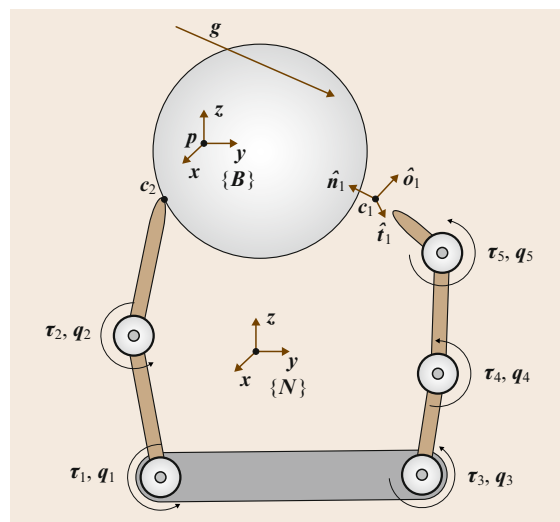


Fig. 28.2 Main quantities for grasp analysis

wrenches see [28.5, 6]. Note that for planar systems, $\mathbf{v} \in \mathbb{R}^2$ and $\omega \in \mathbb{R}$, and so $n_v = 3$.

Another important point is $\dot{\mathbf{u}} \neq \mathbf{v}$. Instead, these variables are related by the matrix \mathbf{V} as:

$$\dot{\mathbf{u}} = \mathbf{V}\mathbf{v}, \quad (28.1)$$

where the matrix $\mathbf{V} \in \mathbb{R}^{n_u \times n_v}$ is not generally square but nonetheless satisfies $\mathbf{V}^\top \mathbf{V} = \mathbf{I}$ [28.7], \mathbf{I} is the identity matrix, and the dot over the \mathbf{u} implies differentiation with respect to time. Note that, for planar systems, $\mathbf{V} = \mathbf{I} \in \mathbb{R}^{3 \times 3}$.

Let $\mathbf{f} \in \mathbb{R}^3$ be the force applied to the object at the point \mathbf{p} and let $\mathbf{m} \in \mathbb{R}^3$ be the applied moment. These are combined into the object load, or wrench, vector denoted by $\mathbf{g} = [\mathbf{f}^\top \mathbf{m}^\top]^\top \in \mathbb{R}^{n_v}$, where \mathbf{f} and \mathbf{m} are expressed in $\{N\}$. Like twists, wrenches can be referred to any convenient frame fixed to the body. One can think of this as translating the line of application of the force until it contains the origin of the new frame, then adjusting the moment component of the wrench to offset the moment induced by moving the line of the force. Last, the force and adjusted moment are expressed in the new frame. As done with the joint loads, the object wrench will be partitioned into two main parts: contact and noncontact wrenches. Throughout this chapter, \mathbf{g} will denote the noncontact wrench on the object.

28.2.1 Velocity Kinematics

The material in this chapter is valid for a wide range of robot hands and other grasping mechanisms. The hand is assumed to be composed of a palm that serves as the common base for any number of fingers, each with any number of joints. The formulations given in this chapter are expressed explicitly in terms of only revolute and prismatic joints. However, most other common joints can be modeled by combinations of revolute and prismatic joints (e.g., cylindrical, spherical, and planar). Any number of contacts may occur between any link and the object.

Grasp Matrix and Hand Jacobian

Two matrices are of the utmost importance in grasp analysis: the *grasp matrix* \mathbf{G} and the *hand Jacobian* \mathbf{J} . These matrices define the relevant velocity kinematics and force transmission properties of the contacts. The following derivations of \mathbf{G} and \mathbf{J} will be done under the assumption that the system is three-dimensional. Changes for planar systems will be noted later.

Each contact should be considered as two coincident points: one on the hand and one on the object. The hand

Jacobian maps the joint velocities to the twists of the hand expressed in the contact frames, while the transpose of the grasp matrix refers the object twist to the contact frames. Finger joint motions induce a rigid-body motion in each link of the hand. It is implicit in the terminology, *twists of the hand*, that the twist referred to contact i is the twist of the link involved in contact i . Thus these matrices can be derived from the transforms that change the reference frame of a twist.

To derive the grasp matrix, let ω_{obj}^N denote the angular velocity of the object expressed in $\{N\}$ and let $\mathbf{v}_{i,\text{obj}}^N$, also expressed in $\{N\}$, denote the velocity of the point on the object coincident with the origin of $\{C\}_i$. These velocities can be obtained from the object twist referred to $\{N\}$ as:

$$\begin{pmatrix} \mathbf{v}_{i,\text{obj}}^N \\ \omega_{\text{obj}}^N \end{pmatrix} = \mathbf{P}_i^\top \mathbf{v}, \quad (28.2)$$

where

$$\mathbf{P}_i = \begin{pmatrix} \mathbf{I}_{3 \times 3} & \mathbf{0} \\ \mathbf{S}(\mathbf{c}_i - \mathbf{p}) & \mathbf{I}_{3 \times 3} \end{pmatrix}, \quad (28.3)$$

$\mathbf{I}_{3 \times 3} \in \mathbb{R}^{3 \times 3}$ is the identity matrix, and $\mathbf{S}(\mathbf{c}_i - \mathbf{p})$ is the cross-product matrix, that is, given a three-vector $\mathbf{r} = [r_x r_y r_z]^\top$, $\mathbf{S}(\mathbf{r})$ is defined as:

$$\mathbf{S}(\mathbf{r}) = \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix}.$$

The object twist referred to $\{C\}_i$ is simply the vector on the left-hand side of (28.2) expressed in $\{C\}_i$. Let $\mathbf{R}_i = [\hat{\mathbf{n}}_i \hat{\mathbf{t}}_i \hat{\mathbf{o}}_i] \in \mathbb{R}^{3 \times 3}$ represent the orientation of the i -th contact frame $\{C\}_i$ with respect to the inertial frame (the unit vectors $\hat{\mathbf{n}}_i$, $\hat{\mathbf{t}}_i$, and $\hat{\mathbf{o}}_i$ are expressed in $\{N\}$). Then the object twist referred to $\{C\}_i$ is given as:

$$\mathbf{v}_{i,\text{obj}} = \bar{\mathbf{R}}_i^\top \begin{pmatrix} \mathbf{v}_{i,\text{obj}}^N \\ \omega_{\text{obj}}^N \end{pmatrix}, \quad (28.4)$$

where $\bar{\mathbf{R}}_i = \text{Blockdiag}(\mathbf{R}_i, \mathbf{R}_i) = \begin{pmatrix} \mathbf{R}_i & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_i \end{pmatrix} \in \mathbb{R}^{6 \times 6}$.

Substituting $\mathbf{P}_i^\top \mathbf{v}$ from (28.2) into (28.4) yields the partial grasp matrix $\tilde{\mathbf{G}}_i^\top \in \mathbb{R}^{6 \times 6}$, which maps the object twist from $\{N\}$ to $\{C\}_i$:

$$\mathbf{v}_{i,\text{obj}} = \tilde{\mathbf{G}}_i^\top \mathbf{v}, \quad (28.5)$$

where

$$\tilde{\mathbf{G}}_i^\top = \bar{\mathbf{R}}_i^\top \mathbf{P}_i^\top. \quad (28.6)$$

The hand Jacobian can be derived similarly. Let $\omega_{i,\text{hnd}}^N$ be the angular velocity of the link of the hand touching the object at contact i , expressed in $\{N\}$, and define $v_{i,\text{hnd}}^N$ as the translational velocity of contact i on the hand, expressed in $\{N\}$. These velocities are related to the joint velocities through the matrix Z_i whose columns are the Plücker coordinates of the axes of the joints [28.5, 6]. We have:

$$\begin{pmatrix} v_{i,\text{hnd}}^N \\ \omega_{i,\text{hnd}}^N \end{pmatrix} = Z_i \dot{q}, \quad (28.7)$$

where $Z_i \in \mathbb{R}^{6 \times n_q}$ is defined as:

$$Z_i = \begin{pmatrix} d_{i,1} & \cdots & d_{i,n_q} \\ l_{i,1} & \cdots & l_{i,n_q} \end{pmatrix}, \quad (28.8)$$

with the vectors $d_{i,j}, l_{i,j} \in \mathbb{R}^3$ defined as:

$$d_{i,j} = \begin{cases} \mathbf{0}_{3 \times 1} & \text{if contact } i \text{ does not affect joint } j, \\ \hat{z}_j & \text{if joint } j \text{ is prismatic,} \\ S(c_i - \zeta_j)^\top \hat{z}_j & \text{if joint } j \text{ is revolute,} \end{cases}$$

$$l_{i,j} = \begin{cases} \mathbf{0}_{3 \times 1} & \text{if contact } i \text{ does not affect joint } j, \\ \mathbf{0}_{3 \times 1} & \text{if joint } j \text{ is prismatic,} \\ \hat{z}_j & \text{if joint } j \text{ is revolute,} \end{cases}$$

where ζ_j is the origin of the coordinate frame associated with the j -th joint and \hat{z}_j is the unit vector in the direction of the z -axis in the same frame, as shown in Fig. 28.11. Both vectors are expressed in $\{N\}$. These frames may be assigned by any convenient method, for example, the Denavit–Hartenberg method [28.8]. The \hat{z}_j -axis is the rotational axis for revolute joints and the direction of translation for prismatic joints.

The final step in referring the hand twists to the contact frames is to change the frame of expression of $v_{i,\text{hnd}}^N$ and $\omega_{i,\text{hnd}}^N$ to $\{C\}_i$:

$$v_{i,\text{hnd}} = \bar{R}_i^\top \begin{pmatrix} v_{i,\text{hnd}}^N \\ \omega_{i,\text{hnd}}^N \end{pmatrix}. \quad (28.9)$$

Combining (28.9) and (28.7) yields the partial hand Jacobian $\tilde{J}_i \in \mathbb{R}^{6 \times n_q}$, which relates the joint velocities to the contact twists on the hand:

$$v_{i,\text{hnd}} = \tilde{J}_i \dot{q}, \quad (28.10)$$

where

$$\tilde{J}_i = \bar{R}_i^\top Z_i. \quad (28.11)$$

To compact notation, stack all the twists of the hand and object into the vectors $v_{c,\text{hnd}} \in \mathbb{R}^{6n_c}$ and $v_{c,\text{obj}} \in \mathbb{R}^{6n_c}$ as follows:

$$v_{c,\xi} = \begin{pmatrix} v_{1,\xi}^\top & \cdots & v_{n_c,\xi}^\top \end{pmatrix}^\top, \quad \xi = \{\text{obj}, \text{hnd}\}.$$

Now the *complete grasp matrix* $\tilde{G} \in \mathbb{R}^{6 \times 6n_c}$ and the *complete hand Jacobian* $\tilde{J} \in \mathbb{R}^{6n_c \times n_q}$ relate the various velocity quantities as

$$v_{c,\text{obj}} = \tilde{G}^\top v, \quad (28.12)$$

$$v_{c,\text{hnd}} = \tilde{J} \dot{q}, \quad (28.13)$$

where

$$\tilde{G}^\top = \begin{pmatrix} \tilde{G}_1^\top \\ \vdots \\ \tilde{G}_{n_c}^\top \end{pmatrix}, \quad \tilde{J} = \begin{pmatrix} \tilde{J}_1 \\ \vdots \\ \tilde{J}_{n_c} \end{pmatrix}. \quad (28.14)$$

The term *complete* is used to emphasize that all $6n_c$ twist components at the contacts are included in the mapping. See Example 1, Part 1 and Example 3, Part 1 at the end of this chapter for clarification.

Contact Modeling

Three contact models useful for grasp analysis are reviewed here. For a complete discussion of contact modeling in robotics, readers are referred to Chap. 27.

The three models of greatest interest in grasp analysis are known as *point contact without friction*, *hard finger*, and *soft finger* [28.9]. These models select components of the contact twists to transmit between the hand and the object. This is done by equating a subset of the components of the hand and object twist at each contact. The corresponding components of the contact force and moment are also equated, but without regard for the constraints imposed by contact unilaterality and friction models (Sect. 28.5.2).

The point-contact-without-friction (**PwoF**) model is used when the contact patch is very small and the surfaces of the hand and object are slippery. With this model, only the normal component of the translational velocity of the contact point on the hand (i.e., the first component of $v_{i,\text{hnd}}$) is transmitted to the object. The two components of tangential velocity and the three components of angular velocity are not transmitted. Analogously, the normal component of the contact force is transmitted, but the frictional forces and moments are assumed to be negligible.

Table 28.1 Primary notation for grasp analysis

Notation	Definition
n_c	number of contacts
n_q	number of joints of hand
n_v	number of degrees of freedom of object
$\mathbf{q} \in \mathbb{R}^{n_q}$	joint displacements
$\dot{\mathbf{q}} \in \mathbb{R}^{n_q}$	joint velocities
$\boldsymbol{\tau} \in \mathbb{R}^{n_q}$	noncontact joint loads
$\mathbf{u} \in \mathbb{R}^{n_u}$	position and orientation of object
$\mathbf{v} \in \mathbb{R}^{n_v}$	twist of object
$\mathbf{g} \in \mathbb{R}^{n_v}$	noncontact object wrench
$\{B\}$	frame fixed in object
$\{C\}_i$	frame at contact i
$\{N\}$	inertial frame

A hard-finger (HF) model is used when there is significant contact friction, but the contact patch is small, so that no appreciable friction moment exists. When this model is applied to a contact, all three translational velocity components of the contact point on the hand (i.e., the first three components of $\mathbf{v}_{i,\text{hnd}}$) and all three components of the contact force are transmitted through the contact. None of the angular velocity components or moment components are transmitted.

The soft-finger (SF) model is used in situations in which the surface friction and the contact patch are large enough to generate significant friction forces and a friction moment about the contact normal. At a contact where this model is enforced, the three translational velocity components of the contact on the hand and the angular velocity component about the contact normal are transmitted (i.e., the first four components of $\mathbf{v}_{i,\text{hnd}}$). Similarly, all three components of contact force and the normal component of the contact moment are transmitted.

Remark. The reader may see a contradiction between the rigid-body assumption and the soft-finger model. The rigid-body assumption is an approximation that simplifies all aspects of the analysis of grasping, but nonetheless it is sufficiently accurate in many real situations and grasp analysis would be impractical without. On the other hand, the need for a soft-finger model is a clear admission that the finger links and object are not rigid. However, it can be usefully applied in situations in which the amount of deformation required to obtain a large contact patch is small. Such situations occur when the local surface geometries are similar. If large finger or body deformations exist in the real sys-

Table 28.2 Selection matrices for three contact models

Model	ℓ_i	\mathbf{H}_{iF}	\mathbf{H}_{iM}
PwoF	1	$\begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$	vacuous
HF	3	$\mathbf{I}_{3 \times 3}$	vacuous
SF	4	$\mathbf{I}_{3 \times 3}$	$\begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$

tem, the rigid-body approach presented in this chapter should be used with caution.

To develop the PwoF, HF, and SF models, define the relative twist at contact i as:

$$\begin{pmatrix} \dot{\mathbf{J}}_i & -\tilde{\mathbf{G}}_i^\top \end{pmatrix} \begin{pmatrix} \dot{\mathbf{q}} \\ \mathbf{v} \end{pmatrix} = \mathbf{v}_{i,\text{hnd}} - \mathbf{v}_{i,\text{obj}}.$$

A particular contact model is defined through the matrix $\mathbf{H}_i \in \mathbb{R}^{\ell_i \times 6}$, which selects ℓ_i components of the relative contact twist and sets them to zero:

$$\mathbf{H}_i(\mathbf{v}_{i,\text{hnd}} - \mathbf{v}_{i,\text{obj}}) = \mathbf{0}.$$

These components are referred to as transmitted degrees of freedom (DOF). Define \mathbf{H}_i as:

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{H}_{iF} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_{iM} \end{bmatrix}, \quad (28.15)$$

where \mathbf{H}_{iF} and \mathbf{H}_{iM} are the translational and rotational component selection matrices, respectively. Table 28.2 gives the definitions of the selection matrices for the three contact models, where *vacuous* means that the corresponding block row matrix in (28.15) is void (i.e., it has zero rows and columns). Notice that, for the SF model, \mathbf{H}_{iM} selects rotation about the contact normal.

After choosing a transmission model for each contact, the contact constraint equations for all n_c contacts can be written in compact form as:

$$\mathbf{H}(\mathbf{v}_{c,\text{hnd}} - \mathbf{v}_{c,\text{obj}}) = \mathbf{0}, \quad (28.16)$$

where

$$\mathbf{H} = \text{Blockdiag}(\mathbf{H}_1, \dots, \mathbf{H}_{n_c}) \in \mathbb{R}^{\ell \times 6n_c},$$

and the number of twist components ℓ transmitted through the n_c contacts is given by $\ell = \sum_{i=1}^{n_c} \ell_i$.

Finally, by substituting (28.12) and (28.13) into (28.16) one obtains:

$$\begin{pmatrix} \mathbf{J} & -\mathbf{G}^\top \end{pmatrix} \begin{pmatrix} \dot{\mathbf{q}} \\ \mathbf{v} \end{pmatrix} = \mathbf{0}, \quad (28.17)$$

where the grasp matrix and hand Jacobian are

$$\begin{aligned} \mathbf{G}^\top &= \mathbf{H}\tilde{\mathbf{G}}^\top \in \mathbb{R}^{\ell \times 6}, \\ \mathbf{J} &= \mathbf{H}\tilde{\mathbf{J}} \in \mathbb{R}^{\ell \times n_q}. \end{aligned} \quad (28.18)$$

Table 28.3 Selection matrices for planar contact models

Model	ℓ_i	H_{iF}	H_{iM}
PwoF	1	$\begin{pmatrix} 1 & 0 \end{pmatrix}$	vacuous
HF/SF	2	$I_{2 \times 2}$	vacuous

For more details on the construction of \mathbf{H} , the grasp matrix, and the hand Jacobian, readers are referred to [28.10–12] and the references therein.

See Example 1, Part 2 and Example 3, Part 2.

It is worth noting that (28.17) can be written in the following form:

$$\mathbf{J}\dot{\mathbf{q}} = \mathbf{v}_{cc,hnd} = \mathbf{v}_{cc,obj} = \mathbf{G}^\top \mathbf{v}, \quad (28.19)$$

where $\mathbf{v}_{cc,hnd}$ and $\mathbf{v}_{cc,obj}$ contain only the components of the twists that are transmitted by the contacts. Note that this equation implies that grasp *maintenance* is defined as the situation in which all these equations are maintained over time. Thus, when a contact is frictionless, contact maintenance implies continued contact, but sliding is allowed. However, when a contact is of the HF type, contact maintenance implies sticking contact, since sliding would violate the HF model. Similarly, for an SF contact, there may be no sliding or relative rotation about the contact normal.

For the remainder of this chapter, it will be assumed that $\mathbf{v}_{cc,hnd} = \mathbf{v}_{cc,obj}$, so the notation will be shortened to \mathbf{v}_{cc} .

Planar Simplifications

Assume that the plane of motion is the (x, y) -plane of $\{N\}$. The vectors \mathbf{v} and \mathbf{g} reduce in dimension from six to three by dropping components three, four, and five. The dimensions of the vectors \mathbf{c}_i and \mathbf{p} reduce from three to two. The i -th rotation matrix becomes $\mathbf{R}_i = (\hat{\mathbf{n}}_i \ \hat{\mathbf{t}}_i) \in \mathbb{R}^{2 \times 2}$ (where the third components of $\hat{\mathbf{n}}_i$ and $\hat{\mathbf{t}}_i$ are dropped) and (28.4) holds with $\bar{\mathbf{R}}_i = \text{Blockdiag}(\mathbf{R}_i, 1) \in \mathbb{R}^{3 \times 3}$. Equation (28.2) holds with:

$$\mathbf{P}_i = \begin{pmatrix} I_{2 \times 2} & \mathbf{0} \\ S_2(\mathbf{c}_i - \mathbf{p}) & 1 \end{pmatrix},$$

where S_2 is the analog of the cross-product matrix for two-dimensional vectors, given as

$$S_2(\mathbf{r}) = \begin{pmatrix} -r_y & r_x \end{pmatrix}.$$

Table 28.4 Vectors of contact force and moment components, also known as the wrench intensity vector, transmitted through contact i

Model	λ_i
PwoF	(f_{in})
HF	$(f_{in} \ f_{it} \ f_{io})^\top$
SF	$(f_{in} \ f_{it} \ f_{io} \ m_{in})^\top$

Equation (28.7) holds with $\mathbf{d}_{i,j} \in \mathbb{R}^2$ and $\mathbf{l}_{i,j} \in \mathbb{R}$ defined as

$$\mathbf{d}_{i,j} = \begin{cases} \mathbf{0}_{2 \times 1} & \text{if contact force } i \text{ does} \\ & \text{not affect the joint } j, \\ \hat{\mathbf{z}}_j & \text{if joint } j \text{ is prismatic,} \\ S(\mathbf{c}_i - \boldsymbol{\zeta}_j)^\top & \text{if joint } j \text{ is revolute,} \end{cases}$$

$$\mathbf{l}_{i,j} = \begin{cases} 0 & \text{if contact force } i \text{ does} \\ & \text{not affect joint } j, \\ 0 & \text{if joint } j \text{ is prismatic,} \\ 1 & \text{if joint } j \text{ is revolute.} \end{cases}$$

The complete grasp matrix and hand Jacobian have reduced sizes: $\tilde{\mathbf{G}}^\top \in \mathbb{R}^{3n_c \times 3}$ and $\tilde{\mathbf{J}} \in \mathbb{R}^{3n_c \times n_q}$.

As far as contact constraint is concerned, (28.15) holds with \mathbf{H}_{iF} and \mathbf{H}_{iM} in Table 28.3.

In planar cases, the SF and HF models are equivalent, because the object and the hand lie in a plane. Rotations about the contact normals would cause out-of-plane motions. Finally, the dimensions of the grasp matrix and hand Jacobian are reduced to the following sizes: $\mathbf{G}^\top \in \mathbb{R}^{\ell \times 3}$ and $\mathbf{J} \in \mathbb{R}^{\ell \times n_q}$. See Example 1, Part 3 and Example 2, Part 1.

28.2.2 Dynamics and Equilibrium

The dynamic equations of the system can be written as:

$$\begin{aligned} \mathbf{M}_{hnd}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}_{hnd}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{J}^\top \boldsymbol{\lambda} &= \boldsymbol{\tau}_{app} \\ \mathbf{M}_{obj}(\mathbf{u})\ddot{\mathbf{u}} + \mathbf{b}_{obj}(\mathbf{u}, \dot{\mathbf{u}}) - \mathbf{G}\boldsymbol{\lambda} &= \mathbf{g}_{app} \end{aligned} \quad (28.20)$$

subject to constraint (28.17),

where $\mathbf{M}_{hnd}(\cdot)$ and $\mathbf{M}_{obj}(\cdot)$ are symmetric positive-definite inertia matrices and $\mathbf{b}_{hnd}(\cdot, \cdot)$ and $\mathbf{b}_{obj}(\cdot, \cdot)$ are the velocity-product terms, \mathbf{g}_{app} is the force and moment applied to the object by gravity and other external sources, $\boldsymbol{\tau}_{app}$ is the vector of external loads and actuator actions, and the vector $\mathbf{G}\boldsymbol{\lambda}$ is the total

wrench applied to the object by the hand. The vector λ contains the contact force and moment components transmitted through the contacts and expressed in the contact frames. Specifically, $\lambda = [\lambda_1^\top \cdots \lambda_{n_c}^\top]^\top$, where $\lambda_i = \mathbf{H}_i[f_{in} \ f_{it} \ f_{io} \ m_{in} \ m_{it} \ m_{io}]^\top$. The subscripts indicate one normal (n) and two tangential (t, o) components of contact force \mathbf{f} and moment \mathbf{m} . For an SF, HF, or PwoF contact, λ_i is defined as in Table 28.4. Finally, it is worth noting that $\mathbf{G}_i \lambda_i = \tilde{\mathbf{G}}_i \mathbf{H}_i \lambda_i$ is the wrench applied through contact i , where $\tilde{\mathbf{G}}_i$ and \mathbf{H}_i are defined in (28.6) and (28.15). The vector λ_i is known as the wrench intensity vector for contact i .

Equation (28.20) represents the dynamics of the hand and object without regard for the kinematic constraints imposed by the contact models. Enforcing them, the dynamic model of the system can be written as:

$$\begin{pmatrix} \mathbf{J}^\top \\ -\mathbf{G} \end{pmatrix} \lambda = \begin{pmatrix} \boldsymbol{\tau} \\ \mathbf{g} \end{pmatrix} \quad (28.21)$$

subject to $\mathbf{J}\dot{\mathbf{q}} = \mathbf{G}^\top \mathbf{v} = \mathbf{v}_{cc}$, where

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\tau}_{\text{app}} - \mathbf{M}_{\text{hnd}}(\mathbf{q})\ddot{\mathbf{q}} - \mathbf{b}_{\text{hnd}}(\mathbf{q}, \dot{\mathbf{q}}), \\ \mathbf{g} &= \mathbf{g}_{\text{app}} - \mathbf{M}_{\text{obj}}(\mathbf{u})\ddot{\mathbf{v}} - \mathbf{b}_{\text{obj}}(\mathbf{u}, \dot{\mathbf{v}}). \end{aligned} \quad (28.22)$$

One should notice that the dynamic equations are closely related to the kinematic model in (28.17). Specifically, just as \mathbf{J} and \mathbf{G}^\top transmit only selected

components of contact twists, \mathbf{J}^\top and \mathbf{G} in (28.20) serve to transmit only the corresponding components of the contact wrenches.

When the inertia terms are negligible, as occurs during slow motion, the system is said to be quasistatic. In this case, (28.22) becomes:

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\tau}_{\text{app}}, \\ \mathbf{g} &= \mathbf{g}_{\text{app}}, \end{aligned} \quad (28.23)$$

and does not depend on joint and object velocities. Consequently, when the grasp is in static equilibrium or moves quasistatically, one can solve the first equation and the constraint in (28.21) independently to compute λ , $\dot{\mathbf{q}}$, and \mathbf{v} . It is worth noting that such a force/velocity decoupled solution is not possible when dynamic effects are appreciable, since the first equation in (28.21) depends on the third one through (28.22).

Remark. Equation (28.21) highlights an important alternative view of the grasp matrix and the hand Jacobian. \mathbf{G} can be thought of as a mapping from the transmitted contact forces and moments to the set wrenches that the hand can apply to the object, while \mathbf{J}^\top can be thought of as a mapping from the transmitted contact forces and moments to the vector of joint loads. Notice that these interpretations hold for both dynamic and quasistatic conditions.

28.3 Controllable Twists and Wrenches

In hand design and in grasp and manipulation planning, it is important to know the set of twists that can be imparted to the object by movements of the fingers, and conversely, the conditions under which the hand can prevent all possible motions of the object. The dual view is that one needs to know the set of wrenches that the hand can apply to the object and under what conditions any wrench in \mathbb{R}^6 can be applied through the contacts. This knowledge will be gained by studying the various subspaces associated with \mathbf{G} and \mathbf{J} [28.13].

The spaces, shown in Fig. 28.3, are the column spaces and null spaces of \mathbf{G} , \mathbf{G}^\top , \mathbf{J} , and \mathbf{J}^\top . Column space (also known as range) and null space will be denoted by $\mathcal{R}(\cdot)$ and $\mathcal{N}(\cdot)$, respectively. The arrows show the propagation of the various velocity and load quantities through the grasping system. For example, in the left part of Fig. 28.3 it is shown how any vector $\dot{\mathbf{q}} \in \mathbb{R}^{n_q}$ can be decomposed into a sum of two orthogonal vectors in

$\mathcal{R}(\mathbf{J}^\top)$ and in $\mathcal{N}(\mathbf{J})$ and how $\dot{\mathbf{q}}$ is mapped to $\mathcal{R}(\mathbf{J})$ by multiplication by \mathbf{J} .

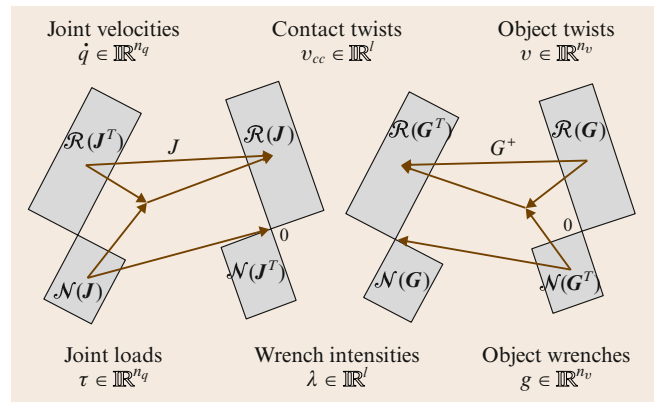


Fig. 28.3 Linear maps relating the twists and wrenches of a grasping system

Table 28.5 Basic grasp classes

Condition	Class	Many to one
$\mathcal{N}(\mathbf{J}) \neq \mathbf{0}$	Redundant	$\dot{\mathbf{q}} \rightarrow \mathbf{v}_{cc}$ $\boldsymbol{\tau} \rightarrow \boldsymbol{\lambda}$
$\mathcal{N}(\mathbf{G}^\top) \neq \mathbf{0}$	Indeterminate	$\mathbf{v} \rightarrow \mathbf{v}_{cc}$ $\mathbf{g} \rightarrow \boldsymbol{\lambda}$ $\boldsymbol{\lambda} \rightarrow \mathbf{g}$
$\mathcal{N}(\mathbf{G}) \neq \mathbf{0}$	Graspable	$\mathbf{v}_{cc} \rightarrow \mathbf{v}$
$\mathcal{N}(\mathbf{J}^\top) \neq \mathbf{0}$	Defective	$\boldsymbol{\lambda} \rightarrow \boldsymbol{\tau}$ $\mathbf{v}_{cc} \rightarrow \dot{\mathbf{q}}$

It is important to recall two facts from linear algebra. First, a matrix \mathbf{A} maps vectors from $\mathcal{R}(\mathbf{A}^\top)$ to $\mathcal{R}(\mathbf{A})$ in a one-to-one and onto fashion, that is, the map \mathbf{A} is a bijection. The generalized inverse \mathbf{A}^+ of \mathbf{A} is a bijection that maps vectors in the opposite direction [28.14]. Also, \mathbf{A} maps vectors in $\mathcal{N}(\mathbf{A})$ to zero. Finally, there is no nontrivial vector that \mathbf{A} can map into $\mathcal{N}(\mathbf{A}^\top)$. This implies that, if $\mathcal{N}(\mathbf{G}^\top)$ is nontrivial, then the hand will not be able to control all degrees of freedom of the object's motion. This is certainly true for quasistatic grasping, but when dynamics are important, they may cause the object to move along the directions in $\mathcal{N}(\mathbf{G}^\top)$.

28.3.1 Grasp Classifications

The four null spaces motivate a basic classification of grasping systems defined in Table 28.5. Assuming solutions to (28.21) exist, the following force and velocity equations provide insight into the physical meaning of the various null spaces:

$$\dot{\mathbf{q}} = \mathbf{J}^+ \mathbf{v}_{cc} + \mathbf{N}(\mathbf{J})\boldsymbol{\alpha}, \quad (28.24)$$

$$\mathbf{v} = (\mathbf{G}^\top)^+ \mathbf{v}_{cc} + \mathbf{N}(\mathbf{G}^\top)\boldsymbol{\beta}, \quad (28.25)$$

$$\boldsymbol{\lambda} = -\mathbf{G}^+ \mathbf{g} + \mathbf{N}(\mathbf{G})\boldsymbol{\gamma}, \quad (28.26)$$

$$\boldsymbol{\lambda} = (\mathbf{J}^\top)^+ \boldsymbol{\tau} + \mathbf{N}(\mathbf{J}^\top)\boldsymbol{\eta}. \quad (28.27)$$

In these equations, \mathbf{A}^+ denotes the generalized inverse, henceforth pseudoinverse, of a matrix \mathbf{A} , $\mathbf{N}(\mathbf{A})$ denotes a matrix whose columns form a basis for $\mathcal{N}(\mathbf{A})$, and $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $\boldsymbol{\gamma}$, and $\boldsymbol{\eta}$ are arbitrary vectors that parameterize the solution sets. If not otherwise specified, the context will make clear whether the generalized inverse is left or right.

If the null spaces represented in the equations are nontrivial, then one immediately sees the first many-to-one mapping in the Table 28.5. To see the other many-to-one mappings, and in particular the defective class, consider (28.24). It can be rewritten with \mathbf{v}_{cc} de-

composed into components \mathbf{v}_{rs} and \mathbf{v}_{lms} in $\mathcal{R}(\mathbf{J})$ and $\mathcal{N}(\mathbf{J}^\top)$, respectively, as follows:

$$\dot{\mathbf{q}} = \mathbf{J}^+ (\mathbf{v}_{rs} + \mathbf{v}_{lms}) + \mathbf{N}(\mathbf{J})\boldsymbol{\alpha}. \quad (28.28)$$

Recall that every vector in $\mathcal{N}(\mathbf{A}^\top)$ is orthogonal to every row of \mathbf{A}^+ . Therefore $\mathbf{J}^+ \mathbf{v}_{lms} = \mathbf{0}$. If $\boldsymbol{\alpha}$ and \mathbf{v}_{rs} are fixed in (28.28), then $\dot{\mathbf{q}}$ is unique. Thus it is clear that, if $\mathcal{N}(\mathbf{J}^\top)$ is nontrivial, then a subspace of twists of the hand at the contacts will map to a single joint velocity vector. Applying the same approach to the other three equations (28.25–28.27) yields the other many-to-one mappings listed in Table 28.5.

Equations (28.21) and (28.24–28.27), motivate the following definitions.

Definition 28.1 Redundant

A grasping system is said to be *redundant* if $\mathcal{N}(\mathbf{J})$ is nontrivial.

Joint velocities $\dot{\mathbf{q}}$ in $\mathcal{N}(\mathbf{J})$ are referred to as *internal hand velocities*, since they correspond to finger motions, but do not generate motion of the hand in the constrained directions at the contact points. If the quasistatic model applies, it can be shown that these motions are not influenced by the motion of the object and vice versa.

Definition 28.2 Indeterminate

A grasping system is said to be *indeterminate* if $\mathcal{N}(\mathbf{G}^\top)$ is nontrivial.

Object twists \mathbf{v} in $\mathcal{N}(\mathbf{G}^\top)$ are called *internal object twists*, since they correspond to motions of the object, but do not cause motion of the object in the constrained directions at the contacts. If the static model applies, it can be shown that these twists cannot be controlled by finger motions.

Definition 28.3 Graspable

A grasping system is said to be *graspable* if $\mathcal{N}(\mathbf{G})$ is nontrivial.

Wrench intensities $\boldsymbol{\lambda}$ in $\mathcal{N}(\mathbf{G})$ are referred to as *internal object forces*. These wrenches are *internal* because they do not contribute to the acceleration of the object, i. e., $\mathbf{G}\boldsymbol{\lambda} = \mathbf{0}$. Instead, these wrench intensities affect the tightness of the grasp. Thus, internal wrench intensities play a fundamental role in maintaining grasps that rely on friction (Sect. 28.5.2).

Definition 28.4 Defective

A grasping system is said to be *defective* if $\mathcal{N}(\mathbf{J}^\top)$ is nontrivial.

Wrench intensities λ in $\mathcal{N}(\mathbf{J}^\top)$ are called *internal hand forces*. These forces do not influence the hand joint dynamics given in (28.20). If the static model is considered, it can be easily shown that wrench intensities belonging to $\mathcal{N}(\mathbf{J}^\top)$ cannot be generated by joint actions, but can be resisted by the structure of the hand.

See Example 1, Part 4; Example 2, Part 2; and Example 3, Part 3.

28.3.2 Limitations of Rigid-Body Assumption

The rigid-body dynamics equation (28.20) can be rewritten with Lagrange multipliers associated with the contact constraints as:

$$\mathbf{M}_{\text{dyn}} \begin{pmatrix} \ddot{\mathbf{q}} \\ \dot{\mathbf{v}} \\ \lambda \end{pmatrix} = \begin{pmatrix} \boldsymbol{\tau} - \mathbf{b}_{\text{hnd}} \\ \mathbf{v} - \mathbf{b}_{\text{obj}} \\ \mathbf{b}_c \end{pmatrix}, \quad (28.29)$$

where $\mathbf{b}_c = [\partial(\mathbf{J}\dot{\mathbf{q}})/\partial\mathbf{q}]\dot{\mathbf{q}} - [\partial(\mathbf{G}\mathbf{v})/\partial\mathbf{u}]\dot{\mathbf{u}}$ and

$$\mathbf{M}_{\text{dyn}} = \begin{pmatrix} \mathbf{M}_{\text{hnd}} & \mathbf{0} & \mathbf{J}^\top \\ \mathbf{0} & \mathbf{M}_{\text{obj}} & -\mathbf{G} \\ \mathbf{J} & -\mathbf{G}^\top & \mathbf{0} \end{pmatrix}.$$

In order for this equation to determine the motion of the system completely, it is necessary that the matrix \mathbf{M}_{dyn} be invertible. This case is considered in detail in [28.15], where the dynamics of multifinger manipulation is studied under the hypothesis that the hand Jacobian is full row rank, $\mathcal{N}(\mathbf{J}^\top) = \mathbf{0}$. For all manipulation systems with noninvertible \mathbf{M}_{dyn} , rigid-body dynamics fails to determine the motion and the wrench intensity vector. By observing that:

$$\begin{aligned} \mathcal{N}(\mathbf{M}_{\text{dyn}}) \\ = \{(\ddot{\mathbf{q}}, \dot{\mathbf{v}}, \lambda)^\top | \ddot{\mathbf{q}} = \mathbf{0}, \dot{\mathbf{v}} = \mathbf{0}, \lambda \in \mathcal{N}(\mathbf{J}^\top) \cap \mathcal{N}(\mathbf{G})\}, \end{aligned}$$

the same arguments apply under the quasistatic conditions defined by (28.21) and (28.23). When $\mathcal{N}(\mathbf{J}^\top) \cap \mathcal{N}(\mathbf{G}) \neq \mathbf{0}$, the rigid-body approach fails to solve the first equation in (28.21), thus leaving λ indeterminate.

Definition 28.5 Hyperstatic

A grasping system is said to be *hyperstatic* if $\mathcal{N}(\mathbf{J}^\top) \cap \mathcal{N}(\mathbf{G})$ is nontrivial.

In such systems there are internal forces (Definition 28.3) belonging to $\mathcal{N}(\mathbf{J}^\top)$ that are not controllable as discussed for defective grasps. Rigid-body dynamics is not satisfactory for hyperstatic grasps since the rigid-body assumption leads to undetermined contact wrenches [28.16].

See Example 3, Part 3.

28.3.3 Desirable Properties

For a general-purpose grasping system, there are three main desirable properties: control of the object twist \mathbf{v} , control of object wrench \mathbf{g} , and control of the internal forces. Control of these quantities implies that the hand can deliver the desired \mathbf{v} and \mathbf{g} with specified grip pressure by the appropriate choice of joint velocities and actions. The conditions on \mathbf{J} and \mathbf{G} equivalent to these properties are given in Table 28.6.

We derive the associated conditions in two steps. First, we ignore the structure and configuration of the hand (captured in \mathbf{J}) by assuming that the contact point on the finger can be commanded to move in any direction transmitted by the chosen contact model. An important perspective here is that \mathbf{v}_{cc} is seen as the independent input variable and \mathbf{v} is seen as the output. The dual interpretation is that the actuators can generate any contact force and moment in the constrained directions. Similarly, λ is seen as the input and \mathbf{g} is seen as the output. The preliminary property of interest under this assumption is whether or not the arrangement and types of contacts on the object (captured in \mathbf{G}) are such that a sufficiently dexterous hand could control its fingers so as to impart any twist $\mathbf{v} \in \mathbb{R}^6$ to the object and, similarly, to apply any wrench $\mathbf{g} \in \mathbb{R}^6$ to the object.

All Object Twists Possible

Given a set of contact locations and types, by solving (28.19) for \mathbf{v} or observing the map \mathbf{G} on the right side of Fig. 28.3, one sees that the achievable object twists are those in $\mathcal{R}(\mathbf{G})$. Those in $\mathcal{N}(\mathbf{G}^\top)$ could not be achieved by any hand using the given grasp. Therefore, to achieve any object twist, we must have: $\mathcal{N}(\mathbf{G}^\top) = \mathbf{0}$, or equiva-

Table 28.6 Desirable properties of a grasp

Task requirement	Required conditions
All wrenches possible, \mathbf{g}	$\text{rank}(\mathbf{G}) = n_v$
All twists possible, \mathbf{v}	
Control all wrenches, \mathbf{g}	$\left\{ \begin{array}{l} \text{rank}(\mathbf{G}) = n_v \\ \text{rank}(\mathbf{G}\mathbf{J}) = n_v \end{array} \right.$
Control all twists, \mathbf{v}	
Control all internal forces	$\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^\top) = \mathbf{0}$



Fig. 28.4 The Salisbury hand

lently, $\text{rank}(\mathbf{G}) = n_v$. Any grasp with three non-collinear hard contacts or two distinct soft contacts satisfies this condition.

All Object Wrenches Possible

This case is the dual of the previous case, so we expect the same condition. From (28.21), one immediately obtains the condition $\mathcal{N}(\mathbf{G}^\top) = \mathbf{0}$, so again we have $\text{rank}(\mathbf{G}) = n_v$.

To obtain the conditions needed to control the various quantities of interest, the structure of the hand cannot be ignored. Recall that the only achievable contact twists on the hand are in $\mathcal{R}(\mathbf{J})$, which is not necessarily equal to \mathbb{R}^ℓ .

Control All Object Twists

By solving (28.17) for \mathbf{v} , one sees that, in order to cause any object twist \mathbf{v} by choice of joint velocities $\dot{\mathbf{q}}$, we must have $\mathcal{R}(\mathbf{G}\mathbf{J}) = \mathcal{R}(\mathbf{G})$ and $\mathcal{N}(\mathbf{G}^\top) = \mathbf{0}$. These conditions are equivalent to $\text{rank}(\mathbf{G}\mathbf{J}) = \text{rank}(\mathbf{G}) = n_v$.

Control All Object Wrenches

This property is dual to the previous one. Analysis of (28.21) yields the same conditions: $\text{rank}(\mathbf{G}\mathbf{J}) = \text{rank}(\mathbf{G}) = n_v$.

Control All Internal Forces

Equation (28.20) shows that wrench intensities with no effect on object motion are only those in $\mathcal{N}(\mathbf{G})$. In general, not all the internal forces may be actively controlled by joint actions. In [28.12, 17] it has been shown that all internal forces in $\mathcal{N}(\mathbf{G})$ are controllable if and only if $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^\top) = \mathbf{0}$.

See Example 1, Part 5 and Example 2, Part 3.

Design Considerations of the Salisbury Hand

The Salisbury hand in Fig. 28.4 was designed to have the smallest number of joints that would meet all the task requirements in Table 28.6. Assuming HF contacts, three non-collinear contacts is the minimum number such that $\text{rank}(\mathbf{G}) = n_v = 6$. In this case, \mathbf{G} has six rows and nine columns and the dimension of $\mathcal{N}(\mathbf{G})$ is three [28.1, 4]. The ability to control all internal forces and apply an arbitrary wrench to the object requires that $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^\top) = \mathbf{0}$, so the minimum dimension of the column space of \mathbf{J} is nine. To achieve this, the hand must have at least nine joints, which Salisbury implemented as three fingers, each with three revolute joints.

The intended way to execute a dexterous manipulation task with the Salisbury hand is to grasp the object at three non-collinear points with the fingertips, forming a grasp triangle. To secure the grasp, the internal forces are controlled so that the contact points are maintained without sliding. Dexterous manipulation can be thought of as moving the fingertips to control the positions of the vertices of the grasp triangle.

28.4 Restraint Analysis

The most fundamental requirements in grasping and dexterous manipulation are the abilities to hold an object in equilibrium and control the position and orientation of the grasped object relative to the palm of the hand. The two most useful characterizations of grasp restraint are *force closure* and *form closure*. These names were in use over 125 years ago in the field of machine design to distinguish between joints that required an external force to maintain contact, and those that did not [28.18]. For example, some water wheels had a cylindrical axle that

was laid in a horizontal semicylindrical groove split on either side of the wheel. During operation, the weight of the wheel acted to *close* the groove–axle contacts, hence the term *force closure*. By contrast, if the grooves were replaced by cylindrical holes just long enough to accept the axle, then the contacts would be closed by the geometry (even if the direction of the gravitational force were reversed), hence the term *form closure*.

When applied to grasping, form and force closure have the following interpretations. Assume that a hand



Fig. 28.5 The palm, fingers, wrist, and watch band combine to create a very secure form closure grasp of a TV remote controller

grasping an object has its joint angles locked and its palm fixed in space; then the grasp has *form closure*, or the object is *form closed*, if it is impossible to move the object, even infinitesimally. Under the same conditions, the grasp has *force closure*, or the object is *force closed*, if for any noncontact wrench experienced by the object, contact wrench intensities exist that satisfy (28.20) and are consistent with the constraints imposed by the friction models applicable at the contact points. Notice that all form closure grasps are also force closure grasps. When under form closure, the object cannot move at



Fig. 28.6 This grasp has a force closure grasp appropriate for dexterous manipulation (Image of Shadow Dextrous Hand © Shadow Robot Company 2008)



Fig. 28.7 In the grasp depicted, contact with the ridges on the gasoline cap creates partial form closure in the direction of cap rotation (when screwing it in) and also in the directions of translation perpendicular to the axis of rotation. To achieve complete control over the cap, the grasp achieves force closure over the other three degrees of freedom

all, regardless of the noncontact wrench. Therefore, the hand maintains the object in equilibrium for any external wrench, which is the force closure requirement.

Roughly speaking, form closure occurs when the palm and fingers wrap around the object forming a cage with no *wiggle room* such as the grasp shown in Fig. 28.5. This kind of grasp is also called a *power grasp* [28.19] or an *enveloping grasp* [28.20]. However, force closure is possible with fewer contacts, as shown in Fig. 28.6, but in this case force closure requires the ability to control internal forces. It is also possible for a grasp to have partial form closure, indicating that only a subset of the possible degrees of freedom are restrained by form closure [28.21]. An example of such a grasp is shown in Fig. 28.7. In this grasp, fingertip placement between the ridges around the periphery of the gasoline cap provide form closure against relative rotation about the axis of the helix of the threads and also against translation perpendicular to that axis, but the other three degrees of freedom are restrained through force closure. Strictly speaking, given a grasp of a real object by a human hand, it is impossible to prevent relative motion of the object with respect to the palm due to the compliance of the hand and object. Preventing all motion is possible only if the contacting bodies are rigid, as is assumed in most mathematical models employed in grasp analysis.

28.4.1 Form Closure

To make the notion of form closure precise, introduce a gap function denoted by $\psi_i(\mathbf{u}, \mathbf{q})$ at each of the n_c contact points between the object and the hand. The gap function is zero at each contact, becomes positive if contact breaks, and negative if penetration occurs. The

gap function can be thought of as the distance between the contact points. In general, this function is dependent on the shapes of the contacting bodies. Let $\bar{\mathbf{u}}$ and $\bar{\mathbf{q}}$ represent the configurations of the object and hand for a given grasp; then:

$$\psi_i(\bar{\mathbf{u}}, \bar{\mathbf{q}}) = 0 \quad \forall i = 1, \dots, n_c. \quad (28.30)$$

The form closure condition can now be stated in terms of a differential change $d\mathbf{u}$ of $\bar{\mathbf{u}}$:

Definition 28.6

A grasp $(\bar{\mathbf{u}}, \bar{\mathbf{q}})$ has *form closure* if and only if the following implication holds:

$$\psi(\bar{\mathbf{u}} + d\mathbf{u}, \bar{\mathbf{q}}) \geq \mathbf{0} \Rightarrow d\mathbf{u} = \mathbf{0}, \quad (28.31)$$

where ψ is the n_c -dimensional vector of gap functions with i -th component equal to $\psi_i(\mathbf{u}, \mathbf{q})$. By definition, inequalities between vectors imply that the inequality is applied between corresponding components of the vectors.

Expanding the gap function vector in a Taylor series about $\bar{\mathbf{u}}$ yields infinitesimal form closure tests of various orders. Let ${}^\beta\psi(\mathbf{u}, \mathbf{q})$, $\beta = 1, 2, 3, \dots$ denote the Taylor series approximation truncated after the terms of order β in $d\mathbf{u}$. From (28.30), it follows that the first-order approximation is:

$${}^1\psi(\bar{\mathbf{u}} + d\mathbf{u}, \bar{\mathbf{q}}) = \left. \frac{\partial \psi(\mathbf{u}, \mathbf{q})}{\partial \mathbf{u}} \right|_{(\bar{\mathbf{u}}, \bar{\mathbf{q}})} d\mathbf{u},$$

where $\partial \psi(\mathbf{u}, \mathbf{q}) / \partial \mathbf{u}|_{(\bar{\mathbf{u}}, \bar{\mathbf{q}})}$ denotes the partial derivative of ψ with respect to \mathbf{u} evaluated at $(\bar{\mathbf{u}}, \bar{\mathbf{q}})$. Replacing ψ with its approximation of order β in (28.31) implies three relevant cases of order β :

1. if there exists $d\mathbf{u}$ such that ${}^\beta\psi(\bar{\mathbf{u}} + d\mathbf{u}, \bar{\mathbf{q}})$ has at least one strictly positive component, then the grasp does not have form closure of order β ;
2. if for every nonzero $d\mathbf{u}$, ${}^\beta\psi(\bar{\mathbf{u}} + d\mathbf{u}, \bar{\mathbf{q}})$ has at least one strictly negative component, then the grasp has form closure of order β ;
3. if neither case 1 nor case 2 applies for all ${}^\alpha\psi(\bar{\mathbf{u}} + d\mathbf{u}, \bar{\mathbf{q}}) \forall \alpha \leq \beta$, then higher-order analysis is required to determine the existence of form closure.

Figure 28.8 illustrates form closure concepts using several planar grasps of gray objects by fingers shown as dark disks. The concepts are identical for grasps of three-dimensional objects, but are more clearly illustrated in a plane. The grasp on the left has first-order form closure. Note that first-order form closure only involves

the first derivatives of the distance functions. This implies that the only relevant geometry in first-order form closure are the locations of the contacts and the directions of the contact normals. The grasp in the center has form closure of higher order, with the specific order depending on the degrees of the curves defining the surfaces of the object and fingers in the neighborhoods of the contacts [28.22]. Second-order form closure analysis depends on the curvatures of the two contacting bodies in addition to the geometric information used to analyze first-order form closure. The grasp on the right does not have form closure of any order, because the object can translate horizontally and rotate about its center.

First-Order Form Closure

First-order form closure exists if and only if the following implication holds:

$$\left. \frac{\partial \psi(\mathbf{u}, \mathbf{q})}{\partial \mathbf{u}} \right|_{(\bar{\mathbf{u}}, \bar{\mathbf{q}})} d\mathbf{u} \geq \mathbf{0} \Rightarrow d\mathbf{u} = \mathbf{0}.$$

The first-order form closure condition can be written in terms of the object twist \mathbf{v} :

$$\mathbf{G}_n^\top \mathbf{v} \geq \mathbf{0} \Rightarrow \mathbf{v} = \mathbf{0}, \quad (28.32)$$

where $\mathbf{G}_n^\top = \partial \psi / \partial \mathbf{u} \mathbf{V} \in \mathbb{R}^{n_c \times 6}$. Because the gap functions only quantify distances, the product $\mathbf{G}_n^\top \mathbf{v}$ is the vector of normal components of the instantaneous velocities of the object at the contact points (which must be nonnegative to prevent interpenetration). This in turn implies that the grasp matrix is the one that would result from the assumption that all contacts are of the type Pwof.

An equivalent condition in terms of the contact wrench intensity vector $\lambda_n \in \mathbb{R}^{n_c}$ can be stated as follows. A grasp has first-order form closure if and only

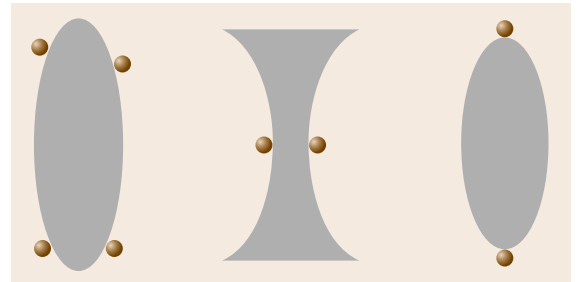


Fig. 28.8 Three planar grasps: two with form closure of different orders and one without form closure

if:

$$\left. \begin{aligned} \mathbf{G}_n \boldsymbol{\lambda}_n &= -\mathbf{g} \\ \boldsymbol{\lambda}_n &\geq \mathbf{0} \end{aligned} \right\} \forall \mathbf{g} \in \mathbb{R}^6. \quad (28.33)$$

The physical interpretation of this condition is that equilibrium can be maintained under the assumption that the contacts are frictionless. Note that the components of $\boldsymbol{\lambda}_n$ are the magnitudes of the normal components of the contact forces. The subscript $(\cdot)_n$ is used to emphasize that $\boldsymbol{\lambda}_n$ contains no other force or moment components.

Since \mathbf{g} must be in the range of \mathbf{G}_n for equilibrium to be satisfied, and since \mathbf{g} is an arbitrary element of \mathbb{R}^6 , then in order for condition (28.33) to be satisfied, the rank of \mathbf{G}_n must be six. Assuming $\text{rank}(\mathbf{G}_n) = 6$, another equivalent mathematical statement of first-order form closure is: there exists $\boldsymbol{\lambda}_n$ such that the following two conditions hold [28.23]:

$$\left. \begin{aligned} \mathbf{G}_n \boldsymbol{\lambda}_n &= \mathbf{0}, \\ \boldsymbol{\lambda}_n &> \mathbf{0}. \end{aligned} \right\} \quad (28.34)$$

This means that there exists a set of strictly compressive normal contact forces in the null space of \mathbf{G}_n . In other words, one can squeeze the object as tightly as desired while maintaining equilibrium. A second interpretation of this condition is that the nonnegative span of the columns of \mathbf{G}_n must equal \mathbb{R}^6 . As will be seen, this interpretation will provide a conceptual link called *frictional form closure* that lies between form closure and force closure.

The duality of conditions (28.32) and (28.33) can be seen clearly by examining the set of wrenches that can be applied by frictionless contacts and the corresponding set of possible object twists. For this discussion, it is useful to give definitions of cones and their duals.

Definition 28.7

A *cone* \mathcal{C} is a set of vectors $\boldsymbol{\zeta}$ such that, for every $\boldsymbol{\zeta}$ in \mathcal{C} , every nonnegative scalar multiple of $\boldsymbol{\zeta}$ is also in \mathcal{C} .

Equivalently, a cone is a set of vectors closed under addition and nonnegative scalar multiplication.

Table 28.7 Minimum number of contacts n_c required to form close an object with n_v degrees of freedom

n_v	n_c
3 (planar grasp)	4
6 (spatial grasp)	7
n_v (general)	$n_v + 1$

Definition 28.8

Given a cone \mathcal{C} with elements $\boldsymbol{\zeta}$, the *dual cone* \mathcal{C}^* with elements $\boldsymbol{\zeta}^*$ is the set of vectors such that the dot product of $\boldsymbol{\zeta}^*$ with each vector in \mathcal{C} is nonnegative. Mathematically:

$$\mathcal{C}^* = \{\boldsymbol{\zeta}^* | \boldsymbol{\zeta}^T \boldsymbol{\zeta}^* \geq 0, \forall \boldsymbol{\zeta} \in \mathcal{C}\}. \quad (28.35)$$

See Example 4.

First-Order Form Closure Requirements

Several useful necessary conditions for form closure are known. In 1897 *Somov* proved that at least seven contacts are necessary to form close a rigid object with six degrees of freedom [28.24]. *Lakshminarayana* generalized this to prove that $n_v + 1$ contacts are necessary to form close an object with n_v degrees of freedom [28.21] (based on *Goldman and Tucker* 1956 [28.25]), see Table 28.7. This led to the definition of partial form closure that was mentioned above in the discussion of the hand grasping the gasoline cap. *Markenscoff* and *Papadimitriou* determined a tight upper bound, showing that, for all objects whose surfaces are not surfaces of revolution, at most $n_v + 1$ contacts are necessary [28.26]. Form closure is impossible to achieve for surfaces of revolution.

To emphasize the fact that $n_v + 1$ contacts are necessary and *not* sufficient, consider grasping a cube with seven or more points of contact. If all contacts are on one face, then clearly the cube is not form closed.

First-Order Form Closure Tests

Because form closure grasps are very secure, it is desirable to design or synthesize such grasps. To do this, one needs a way to test candidate grasps for form closure, and rank them so that the *best* grasp can be chosen. One reasonable measure of form closure can be derived from the geometric interpretation of the condition (28.34). The null space constraint and the positivity of $\boldsymbol{\lambda}_n$ represent the addition of the columns of \mathbf{G}_n scaled by the components of $\boldsymbol{\lambda}_n$. Any choice of $\boldsymbol{\lambda}_n$ closing this loop is in $\mathcal{N}(\mathbf{G}_n)$. For a given loop, if the magnitude of the smallest component of $\boldsymbol{\lambda}_n$ is positive, then the grasp has form closure, otherwise it does not. Let us denote this smallest component by d . Since such a loop, and hence d , can be scaled arbitrarily, $\boldsymbol{\lambda}_n$ should be bounded for computational expediency.

After verifying that \mathbf{G}_n has full row rank, a quantitative form closure test based on the above observations can be formulated as a linear program (LP) in the

unknowns d and λ_n as follows:

$$LPI: \quad \text{maximize :} \quad d \quad (28.36)$$

$$\text{subject to:} \quad \mathbf{G}_n \lambda_n = \mathbf{0} \quad (28.37)$$

$$\mathbf{I} \lambda_n - \mathbf{I} d \geq \mathbf{0} \quad (28.38)$$

$$d \geq 0 \quad (28.39)$$

$$\mathbf{I}^\top \lambda_n \leq n_c, \quad (28.40)$$

where $\mathbf{I} \in \mathbb{R}^{n_c \times n_c}$ is the identity matrix and $\mathbf{I} \in \mathbb{R}^n$ is a vector with all components equal to 1. The last inequality is designed to prevent this LP from becoming unbounded. A typical LP solution algorithm determines infeasibility or unboundedness of the constraints in the, so-called, phase I of the algorithm, and considers the result before attempting to calculate an optimal value [28.27]. If LPI is infeasible, or if the optimal value d^* is zero, then the grasp is not form closed.

The quantitative form closure test (28.36–28.40) has $n_c + 8$ constraints and $n_c + 1$ unknowns. For a typical grasp with $n_c < 10$, this is a small linear program that can be solved very quickly using the simplex method. However, one should note that the metric d^* is dependent on the choice of units used when forming \mathbf{G}_n . It would be advisable to nondimensionalize the components of the wrenches to avoid dependence of the optimal d on ones choice of units. This could be done by dividing the first three rows of \mathbf{G} by a characteristic force and the last three rows by a characteristic moment.

However, if one desires a binary test, LPI can be converted into one by dropping the last constraint (28.40) and applying only phase I of the simplex algorithm.

In summary, form closure testing is a two-step process:

Form Closure Test

1. Compute $\text{rank}(\mathbf{G}_n)$.
 - a) If $\text{rank}(\mathbf{G}_n) \neq n_v$, then form closure does not exist. Stop.
 - b) If $\text{rank}(\mathbf{G}_n) = n_v$, continue.
2. Solve LPI.
 - a) If $d^* = 0$, then form closure does not exist.
 - b) If $d^* > 0$, then form closure exists and d^* is a crude measure of how far the grasp is from losing form closure.

Variations of the Test

If the rank test fails, then the grasp could have partial form closure over as many as $\text{rank}(\mathbf{G}_n)$ degrees of freedom. If one desires to test this, then LPI must be solved using a new \mathbf{G}_n formed by retaining only the rows cor-

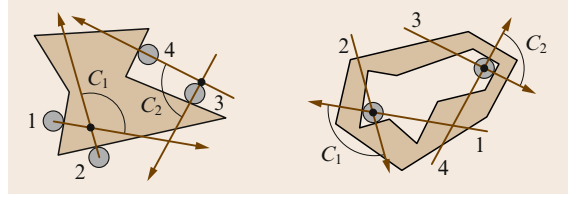


Fig. 28.9 Planar grasps with first-order form closure

responding to the degrees of freedom for which partial form closure is to be tested. If $d^* > 0$, then partial form closure exists. A second variation is to constrain d to be greater than some large negative value. If this is done, then $d^* < 0$ is a crude measure of how far a grasp is from achieving form closure.

See Example 5.

Planar Simplifications

In the planar case, Nguyen [28.28] developed a graphical qualitative test for form closure. Figure 28.9 shows two form closure grasps with four contacts. To test form closure one partitions the normals into two groups of two. Let \mathcal{C}_1 be the nonnegative span of two normals in one pair and \mathcal{C}_2 be the nonnegative span of the other pair. A grasp has form closure if and only if \mathcal{C}_1 and \mathcal{C}_2 or $-\mathcal{C}_1$ and $-\mathcal{C}_2$ see each other for any pairings. Two cones see each other if the open line segment defined by the vertices of the cones lies in the interior of both cones. In the presence of more than four contacts, if any set of four contacts satisfies this condition, then the grasp has form closure. Notice that this graphical test can be difficult to execute for grasps with more than four contacts. Also, it does not extend to grasps of three-dimensional (3-D) objects and does not provide a closure measure.

28.4.2 Force Closure

A grasp has force closure, or is force closed, if the grasp can be maintained in the face of any object wrench. Force closure is similar to form closure, but relaxed to allow friction forces to help balance the object wrench. A benefit of including friction in the analysis is the reduction in the number of contact points needed for closure. A three-dimensional object with six degrees of freedom requires seven contacts for form closure, but for force closure, only two contacts are needed if they are modeled as soft fingers, and only three (non-collinear) contacts are needed if they are modeled as hard fingers.

Force closure relies on the ability of the hand to squeeze arbitrarily tightly in order to compensate for large applied wrenches that can only be resisted by

Table 28.8 Norms for the three main contact models

Model	$\ \lambda_i\ _w$
PwoF	0
HF	$\frac{1}{\mu_i} \sqrt{f_{it}^2 + f_{io}^2}$
SF	$\frac{1}{\mu_i} \sqrt{f_{it}^2 + f_{io}^2} + \frac{1}{av_i} m_{in} $

friction. Figure 28.14 shows a grasped polygon (see Example 2). Consider applying a wrench to the object that is a pure force acting upward along the y -axis of the inertial frame. It seems intuitive that, if there is enough friction, the hand will be able to squeeze the object with friction forces preventing the object's upward escape. Also, as the applied force increases in magnitude, the magnitude of the squeezing force will have to increase accordingly.

Since force closure is dependent on the friction models, common models will be introduced before giving formal definitions of force closure.

Friction Models

Recall the components of force and moment transmitted through contact i under the various contact models given earlier (Table 28.4). At contact point i , the friction law imposes constraints on the components of the contact force and moment. Specifically, the frictional components of λ_i are constrained to lie inside a limit surface, denoted by \mathcal{L}_i , that scales linearly with the product $\mu_i f_{in}$, where μ_i is the coefficient of friction at contact i . In the case of Coulomb friction, the limit surface is a circle of radius $\mu_i f_{in}$. The Coulomb friction cone \mathcal{F}_i is a subset of \mathbb{R}^3 :

$$\mathcal{F}_i = \left\{ (f_{in}, f_{it}, f_{io}) \mid \sqrt{f_{it}^2 + f_{io}^2} \leq \mu_i f_{in} \right\}. \quad (28.41)$$

More generally, the friction laws of interest have limit surfaces defined in the space of friction components, \mathbb{R}^{ℓ_i-1} , and friction cones \mathcal{F}_i defined in the space of λ_i , \mathbb{R}^{ℓ_i} . They can be written as

$$\mathcal{F}_i = \left\{ \lambda_i \in \mathbb{R}^{\ell_i} \mid \|\lambda_i\|_w \leq f_{in} \right\}, \quad (28.42)$$

where $\|\lambda_i\|_w$ denotes a weighted quadratic norm of the friction components at contact i . The limit surface is defined by $\|\lambda_i\|_w = f_{in}$.

Table 28.8 defines useful weighted quadratic norms for the three contact models: PwoF, HF, and SF. The parameter μ_i is the friction coefficient for the tangential forces, v_i is the torsional friction coefficient, and a is the characteristic length of the object that is used to ensure consistent units in the terms of the norm of the SF model.

Remark. There are several noteworthy points to be made about the friction cones. First, all of them implicitly or explicitly constrain the normal component of the contact force to be nonnegative. The cone for SF contacts has a cylindrical limit surface with circular cross section in the (f_{it}, f_{io}) -plane and rectangular cross section in the (f_{it}, m_{in}) -plane. With this model, the amount of torsional friction that can be transmitted is independent of the lateral friction load. An improved model that couples the torsional friction limit with the tangential limit was studied by Howe and Cutkosky [28.29].

A Force Closure Definition

One common definition of force closure can be stated simply by modifying condition (28.33) to allow each contact force to lie in its friction cone rather than along the contact normal. Because this definition does not consider the hand's ability to control contact forces, this definition will be referred to as *frictional form closure*. A grasp will be said to have frictional form closure if and only if the following conditions are satisfied:

$$\left. \begin{array}{l} G\lambda = -g \\ \lambda \in \mathcal{F} \end{array} \right\} \forall g \in \mathbb{R}^{n_v},$$

where \mathcal{F} is the composite friction cone defined as: $\mathcal{F} = \mathcal{F}_1 \times \cdots \times \mathcal{F}_{n_c} = \{\lambda \in \mathbb{R}^m \mid \lambda_i \in \mathcal{F}_i; i = 1, \dots, n_c\}$, and each \mathcal{F}_i is defined by (28.42) and one of the models listed in Table 28.8.

Letting $\text{Int}(\mathcal{F})$ denote the interior of the composite friction cone, Murray et al. give the following equivalent definition [28.15]:

Definition 28.9

(Proposition 5.2, Murray et al.) A grasp has *frictional form closure* if and only if the following conditions are satisfied:

1. $\text{rank}(G) = n_v$
2. $\exists \lambda$ such that $G\lambda = \mathbf{0}$ and $\lambda \in \text{Int}(\mathcal{F})$.

These conditions define what Murray et al. call *force closure*. The force closure definition adopted here is stricter than frictional form closure; it additionally requires that the hand be able to control the internal object forces.

Definition 28.10

A grasp has *force closure* if and only if $\text{rank}(G) = n_v$, $\mathcal{N}(G) \cap \mathcal{N}(J^T) = \mathbf{0}$, and there exists λ such that $G\lambda = \mathbf{0}$ and $\lambda \in \text{Int}(\mathcal{F})$.

The full row rank condition on the matrix \mathbf{G} is the same condition required for form closure, although \mathbf{G} is different from \mathbf{G}_n used to determine form closure. If the rank test passes, then one must still find λ satisfying the remaining three conditions. Of these, the null space intersection test can be performed easily by linear programming techniques, but the friction cone constraint is quadratic, and thus forces one to use nonlinear programming techniques. While exact nonlinear tests have been developed [28.30], only approximate tests will be presented here.

Approximate Force Closure Tests

Any of the friction cones discussed can be approximated as the nonnegative span of a finite number n_g of generators s_{ij} of the friction cone. Given this, one can represent the set of applicable contact wrenches at contact i as follows:

$$\mathbf{G}_i \lambda_i = \mathbf{S}_i \sigma_i, \quad \sigma_i \geq 0,$$

where $\mathbf{S}_i = (s_{i1} \cdots s_{in_g})$ and σ_i is a vector of nonnegative generator weights. If contact i is frictionless, then $n_g = 1$ and $\mathbf{S}_i = [\hat{n}_i^\top (c_i - p) \times \hat{n}_i]^\top]^\top$.

If contact i is of type HF, we represent the friction cone by the nonnegative sum of uniformly spaced contact force generators (Fig. 28.10) whose nonnegative span approximates the Coulomb cone with an inscribed regular polyhedral cone. This leads to the following definition of \mathbf{S}_i :

$$\mathbf{S}_i = \begin{pmatrix} \cdots & 1 & \cdots \\ \cdots & \mu_i \cos(2k\pi/n_g) & \cdots \\ \cdots & \mu_i \sin(2k\pi/n_g) & \cdots \end{pmatrix}, \quad (28.43)$$

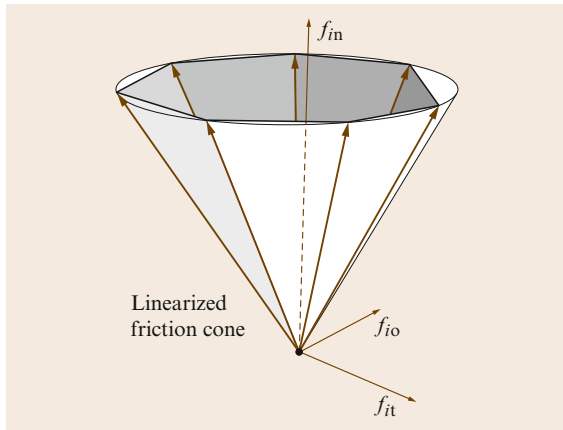


Fig. 28.10 Quadratic cone approximated as a polyhedral cone with seven generators

where the index k varies from 1 to n_g . If one prefers to approximate the quadratic friction cone by a circumscribing polyhedral cone, one simply replaces μ_i in the above definition with $\mu_i / \cos(\pi/n_g)$.

The adjustment needed for the SF model is quite simple. Since the torsional friction in this model is decoupled from the tangential friction, its generators are given by $[1 \ 0 \ 0 \ \pm b v_i]^\top$. Thus \mathbf{S}_i for the SF model is:

$$\mathbf{S}_i = \begin{pmatrix} \cdots & 1 & \cdots & 1 & 1 \\ \cdots & \mu_i \cos(2k\pi/n_g) & \cdots & 0 & 0 \\ \cdots & \mu_i \sin(2k\pi/n_g) & \cdots & 0 & 0 \\ \cdots & 0 & \cdots & b v_i & -b v_i \end{pmatrix}, \quad (28.44)$$

where b is the characteristic length used to unify units. The set of total contact wrenches that may be applied by the hand without violating the contact friction law at any contact can be written as:

$$\mathbf{G} \lambda = \mathbf{S} \sigma, \quad \sigma \geq 0,$$

where $\mathbf{S} = (\mathbf{S}_1, \dots, \mathbf{S}_{n_g})$ and $\sigma = (\sigma_1^\top \cdots \sigma_{n_g}^\top)^\top$.

It is convenient to reformulate the friction constraints in a dual form:

$$\mathbf{F}_i \lambda_i \geq 0. \quad (28.45)$$

In this form, each row of \mathbf{F}_i is normal to a face formed by two adjacent generators of the approximate cone. For an HF contact, row i of \mathbf{F}_i can be computed as the cross product of s_i and s_{i+1} . In the case of an SF contact, the generators are of dimension four, so simple cross products will not suffice. However, general methods exist to perform the conversion from the generator form to the face normal form [28.25].

The face normal constraints for all contacts can be combined into the following compact form:

$$\mathbf{F} \lambda \geq 0, \quad (28.46)$$

where $\mathbf{F} = \text{Blockdiag}(\mathbf{F}_1, \dots, \mathbf{F}_{n_c})$.

Let $e_i \in \mathbb{R}^{\ell_i}$ be the first row of \mathbf{H}_i . Further let $e = (e_1, \dots, e_{n_c}) \in \mathbb{R}^\ell$ and let $\mathbf{E} = \text{Blockdiag}(e_1, \dots, e_{n_c}) \in \mathbb{R}^{\ell \times n_c}$. The following linear program is a quantitative test for frictional form closure. The optimal objective function value d^* is a measure of the distance the contact forces are from the boundaries of their friction cones, and hence a crude measure of how far a grasp is from

losing frictional form closure.

$$\begin{aligned}
 LP2: \quad & \text{maximize :} && d \\
 & \text{subject to:} && \mathbf{G}\boldsymbol{\lambda} = \mathbf{0} \\
 & && \mathbf{F}\boldsymbol{\lambda} - \mathbf{I}d \geq \mathbf{0} \\
 & && d \geq 0 \\
 & && \mathbf{e}\boldsymbol{\lambda} \leq n_c.
 \end{aligned}$$

The last inequality in *LP2* is simply the sum of the magnitudes of the normal components of the contact forces. After solving *LP2*, if $d^* = 0$ frictional form closure does not exist, but if $d^* > 0$, then it does.

If the grasp has frictional form closure, the last step to determine the existence of force closure is to verify the condition $\mathcal{N}(\mathbf{G}) \cap \mathcal{N}(\mathbf{J}^\top) = \mathbf{0}$. If it holds, then the grasp has force closure. This condition is easy to verify with another linear program *LP3*.

$$\begin{aligned}
 LP3: \quad & \text{maximize :} && d \\
 & \text{subject to:} && \mathbf{G}\boldsymbol{\lambda} = \mathbf{0} \\
 & && \mathbf{J}^\top \boldsymbol{\lambda} = \mathbf{0} \\
 & && \mathbf{E}\boldsymbol{\lambda} - \mathbf{I}d \geq \mathbf{0} \\
 & && d \geq 0 \\
 & && \mathbf{e}\boldsymbol{\lambda} \leq n_c.
 \end{aligned}$$

In summary, force closure testing is a three-step process:

Approximate Force Closure Test

1. Compute $\text{rank}(\mathbf{G})$.
 - a) If $\text{rank}(\mathbf{G}) \neq n_v$, then force closure does not exist. Stop.

- b) If $\text{rank}(\mathbf{G}) = n_v$, continue.
2. Solve *LP2: Test frictional form closure*.
 - a) If $d^* = 0$, then frictional form closure does not exist. Stop.
 - b) If $d^* > 0$, then frictional form closure exists and d^* is a crude measure of how far the grasp is from losing frictional form closure.
3. Solve *LP3. Test control of internal force*.
 - a) If $d^* > 0$, then force closure does not exist.
 - b) If $d^* = 0$, then force closure exists.

See Example 1, Part 6.

Planar Simplifications

In planar grasping systems, the approximate method described above is exact. This is because the **SF** models are meaningless, since rotations about the contact normal would cause motions out of the plane. With regard to the **HF** model, for planar problems, the quadratic friction cone becomes linear, with its cone represented exactly as:

$$\mathbf{F}_i = \frac{1}{\sqrt{1 + \mu_i^2}} \begin{pmatrix} \mu_i & 1 \\ \mu_i & -1 \end{pmatrix}. \quad (28.47)$$

Nguyen's graphical form closure test can be applied to planar grasps with two frictional contacts [28.28]. The only change is that the four contact normals are replaced by the four generators of the two friction cones. However, the test can only determine frictional form closure, since it does not incorporate the additional information needed to determine force closure.

28.5 Examples

28.5.1 Example 1: Grasped Sphere

Part 1: $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{J}}$

Figure 28.11 shows a planar projection of a three-dimensional sphere of radius r grasped by two fingers, which make two contacts at angles θ_1 and θ_2 . The frames $\{\mathbf{C}\}_1$ and $\{\mathbf{C}\}_2$ are oriented so that their $\hat{\mathbf{o}}$ -directions point out of the plane of the figure (as indicated by the small bold circle). The axes of the frames $\{\mathbf{N}\}$ and $\{\mathbf{B}\}$ were chosen to be axis-aligned with coincident origins located at the center of the sphere. The z -axes are pointing out of the page. Observe that, since the two joint axes of the left finger are perpendicular to the (x, y) -plane, it operates in that plane for all time. The other finger has three

revolute joints. Because its first and second axes, $\hat{\mathbf{z}}_3$ and $\hat{\mathbf{z}}_4$, currently lie in the plane, rotation about $\hat{\mathbf{z}}_3$ will cause $\hat{\mathbf{z}}_4$ to attain an out-of-plane component and would cause the finger tip at contact 2 to leave the plane.

In the current configuration, the rotation matrix for the i -th contact frame is defined as

$$\mathbf{R}_i = \begin{pmatrix} -\cos(\theta_i) & \sin(\theta_i) & 0 \\ -\sin(\theta_i) & -\cos(\theta_i) & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (28.48)$$

The vector from the origin of $\{\mathbf{N}\}$ to the i -th contact point is given by

$$\mathbf{c}_i - \mathbf{p} = r \begin{pmatrix} \cos(\theta_i) & \sin(\theta_i) & 0 \end{pmatrix}^\top. \quad (28.49)$$

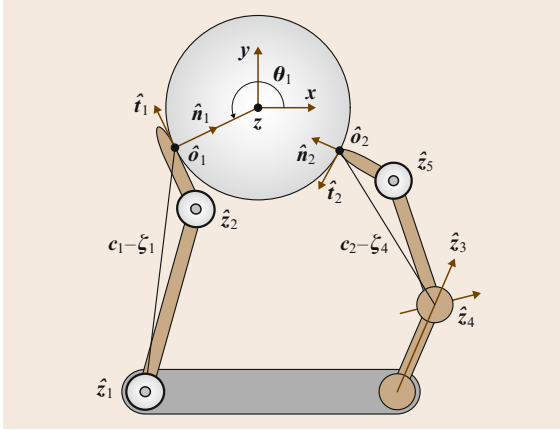


Fig. 28.11 A sphere grasped by a two-fingered hand with five revolute joints

Substituting into (28.3) and (28.6) yields the complete grasp matrix for contact i :

$$\tilde{G}_i = \begin{pmatrix} -c_i & s_i & 0 & & & \\ -s_i & -c_i & 0 & & \mathbf{0} & \\ 0 & 0 & 1 & & & \\ 0 & 0 & rs_i & -c_i & s_i & 0 \\ 0 & 0 & -rc_i & -s_i & -c_i & 0 \\ 0 & -r & 0 & 0 & 0 & 1 \end{pmatrix}, \quad (28.50)$$

where $\mathbf{0} \in \mathbb{R}^{3 \times 3}$ is the zero matrix and c_i and s_i are abbreviations for $\cos(\theta_i)$ and $\sin(\theta_i)$, respectively. The complete grasp matrix is defined as: $\tilde{G} = (\tilde{G}_1 \ \tilde{G}_2) \in \mathbb{R}^{6 \times 12}$.

The accuracy of this matrix can be verified by inspection; for example, the first column is the unit wrench of the unit contact normal, the first three components are the direction cosines of \hat{n}_i , and the last three are $(c_i - p) \times \hat{n}_i$. Since \hat{n}_i is collinear with $(c_i - p)$, the cross products (the last three components of the column) are zero. The last three components of the second column represent the moment of \hat{t}_i about the x -, y -, and z -axes of $\{N\}$. Since \hat{t}_i lies in the (x, y) -plane, the moments with the x - and y -axes are zero. Clearly \hat{t}_i produces a moment of $-r$ about the z -axis.

Construction of the complete hand Jacobian \tilde{J}_i for contact i requires knowledge of the joint axis directions and the origins of the frames fixed to the links of

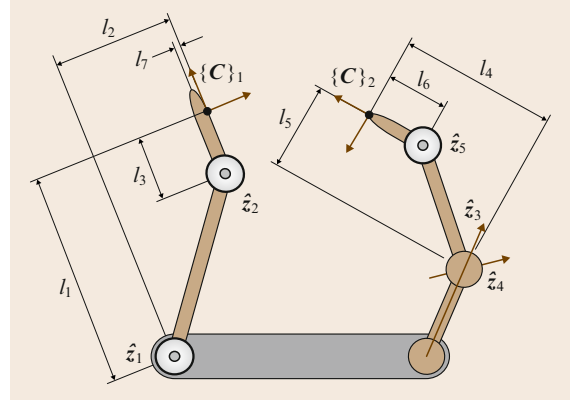


Fig. 28.12 Relevant data for the hand Jacobian

each finger. Figure 28.12 shows the hand in the same configuration as in Fig. 28.11, but with some additional data needed to construct the hand Jacobian. Assume that the origins of the joint frames lie in the plane of the figure.

In the current configuration, the quantities of interest for contact 1, expressed in $\{C\}_1$ are:

$$c_1 - \zeta_1 = \begin{pmatrix} l_2 & l_1 & 0 \end{pmatrix}^T, \quad (28.51)$$

$$c_1 - \zeta_2 = \begin{pmatrix} l_7 & l_3 & 0 \end{pmatrix}^T, \quad (28.52)$$

$$\hat{z}_1 = \hat{z}_2 = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T. \quad (28.53)$$

The quantities of interest for contact 2, in $\{C\}_2$ are:

$$c_2 - \zeta_3 = c_2 - \zeta_4 = \begin{pmatrix} l_4 & -l_5 & 0 \end{pmatrix}^T, \quad (28.54)$$

$$c_2 - \zeta_5 = \begin{pmatrix} l_6 & 0 & 0 \end{pmatrix}^T, \quad (28.55)$$

$$\hat{z}_3 = \begin{pmatrix} 0 & -1 & 0 \end{pmatrix}^T, \quad (28.56)$$

$$\hat{z}_4(q_3) = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 & -1 & 0 \end{pmatrix}^T, \quad (28.57)$$

$$\hat{z}_5(q_3, q_4) = \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^T. \quad (28.58)$$

Generally all of the components of the $c - \zeta$ and \hat{z} vectors (including the components that are zero in the current configuration) are functions of q and u . The dependencies of the \hat{z} vectors are shown explicitly.

Substituting into (28.14), (28.11), and (28.8) yields the complete hand Jacobian $\tilde{\mathbf{J}} \in \mathbb{R}^{12 \times 5}$:

$$\tilde{\mathbf{J}} = \left(\begin{array}{cc|cccc} -l_1 & -l_3 & & & & \\ l_2 & l_7 & & & & \\ 0 & 0 & & & & \\ 0 & 0 & & & \mathbf{0} & \\ 0 & 0 & & & & \\ 1 & 1 & & & & \\ \hline & & 0 & 0 & 0 & \\ & & 0 & 0 & l_6 & \\ & & l_4 & \frac{\sqrt{2}}{2}(l_4 + l_5) & 0 & \\ \mathbf{0} & & 0 & -\frac{\sqrt{2}}{2} & 0 & \\ & & -1 & -\frac{\sqrt{2}}{2} & 0 & \\ & & 0 & 0 & 1 & \end{array} \right).$$

The horizontal dividing line partitions $\tilde{\mathbf{J}}$ into $\tilde{\mathbf{J}}_1$ (on top) and $\tilde{\mathbf{J}}_2$ (on the bottom). The columns correspond to joints 1–5. The block diagonal structure is a result of the fact that finger i directly affects only contact i .

Example 1, Part 2: \mathbf{G} and \mathbf{J}

Assume that the contacts in Fig. 28.11 are both of type SF. Then the selection matrix \mathbf{H} is given by

$$\mathbf{H} = \left(\begin{array}{cc|cccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \hline & & & & 1 & 0 & 0 & 0 & 0 & 0 \\ & & & & 0 & 1 & 0 & 0 & 0 & 0 \\ \mathbf{0} & & & & 0 & 0 & 1 & 0 & 0 & 0 \\ & & & & 0 & 0 & 0 & 1 & 0 & 0 \end{array} \right),$$

thus the matrices $\mathbf{G}^\top \in \mathbb{R}^{8 \times 6}$ and $\mathbf{J} \in \mathbb{R}^{8 \times 5}$ are constructed by removing rows 5, 6, 11, and 12 from $\tilde{\mathbf{G}}^\top$ and $\tilde{\mathbf{J}}$:

$$\mathbf{G}^\top = \left(\begin{array}{cccccc|cccc} -c_1 & -s_1 & 0 & 0 & 0 & 0 & & & & \\ s_1 & -c_1 & 0 & 0 & 0 & -r & & & & \\ 0 & 0 & 1 & rs_1 & -rc_1 & 0 & & & & \\ 0 & 0 & 0 & -c_1 & -s_1 & 0 & & & & \\ \hline -c_2 & -s_2 & 0 & 0 & 0 & 0 & & & & \\ s_2 & -c_2 & 0 & 0 & 0 & -r & & & & \\ 0 & 0 & 1 & rs_2 & -rc_2 & 0 & & & & \\ 0 & 0 & 0 & -c_2 & -s_2 & 0 & & & & \end{array} \right), \quad (28.59)$$

$$\mathbf{J} = \left(\begin{array}{cc|ccc} -l_1 & -l_3 & & & \\ l_2 & l_7 & & & \mathbf{0} \\ 0 & 0 & & & \\ 0 & 0 & & & \\ \hline & & 0 & 0 & 0 \\ & & 0 & 0 & l_6 \\ \mathbf{0} & & d & \frac{\sqrt{2}}{2}(l_4 + l_5) & 0 \\ & & 0 & -\frac{\sqrt{2}}{2} & 0 \end{array} \right). \quad (28.60)$$

Notice that changing the contact models is easily accomplished by removing more rows. Changing contact 1 to HF would eliminate the fourth rows from \mathbf{G}^\top and \mathbf{J} , while changing it to PwoF would eliminate the second, third, and fourth rows of \mathbf{G}^\top and \mathbf{J} . Changing the model at contact 2 would remove either just the eighth row or the sixth, seventh, and eighth rows.

Example 1, Part 3: Reduction to the Planar Case

The grasp shown in Fig. 28.11 can be reduced to a planar problem by following the explicit formulas given above, but it can also be done by understanding the physical interpretations of the various rows and columns of the matrices. Proceed by eliminating velocities and forces that are out of the plane. This can be done by removing the z -axes from $\{\mathbf{N}\}$ and $\{\mathbf{B}\}$, and the \hat{o} -directions at the contacts. Further, joints 3 and 4 must be locked. The resulting \mathbf{G}^\top and \mathbf{J} are constructed eliminating certain rows and columns. \mathbf{G}^\top is formed by removing rows 3, 4, 7, and 8 and columns 3, 4, and 5. \mathbf{J} is formed by removing rows 3, 4, 7, and 8 and columns 3 and 4,

Table 28.9 Dimensions of the main subspaces and classifications of the grasp studied in Example 1

Models	Dimension	Class
HF, HF	$\dim \mathcal{N}(\mathbf{J}) = 1$	Redundant
	$\dim \mathcal{N}(\mathbf{G}^\top) = 1$	Indeterminate
	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^\top) = 2$	Defective
SF, HF	$\dim \mathcal{N}(\mathbf{J}) = 1$	Redundant
	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^\top) = 3$	Defective
HF, SF	$\dim \mathcal{N}(\mathbf{G}) = 1$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^\top) = 2$	Defective
SF, SF	$\dim \mathcal{N}(\mathbf{G}) = 2$	Graspable
	$\dim \mathcal{N}(\mathbf{J}^\top) = 3$	Defective

yielding:

$$\mathbf{G}^\top = \begin{pmatrix} -c_1 & -s_1 & 0 \\ s_1 & -c_1 & -r \\ -c_2 & -s_2 & 0 \\ s_2 & -c_2 & -r \end{pmatrix}, \quad (28.61)$$

$$\mathbf{J} = \begin{pmatrix} -l_1 & -l_3 & 0 \\ l_2 & l_7 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & l_6 \end{pmatrix}. \quad (28.62)$$

Example 1, Part 4: Grasp Classes

The first column of Table 28.9 reports the dimensions of the main subspaces of \mathbf{J} and \mathbf{G} for the sphere grasping example with different contact models. Only nontrivial null spaces are listed.

In the case of two HF contact models, all four null spaces are nontrivial, so the system satisfies the conditions for all four grasp classes. The system is graspable because there is an internal force along the line segment connecting the two contact points. Indeterminacy is manifested in the fact that the hand cannot resist a moment acting about that line. Redundancy is seen to exist since joint 3 can be used to move contact 2 out of the plane of the figure, but joint 4 can be rotated in the opposite direction to cancel this motion. Finally, the grasp is defective, because the contact forces and the instantaneous velocities along the $\hat{\mathbf{d}}_1$ and $\hat{\mathbf{n}}_2$ directions of contact 1 and 2, respectively, cannot be controlled through the joint torques and velocities. These interpretations are borne out in the null space basis matrices below, computed using $r = 1$, $\cos(\theta_1) = -0.8 = -\cos(\theta_2)$, $\sin(\theta_1) = \cos(\theta_2) = -0.6$, and $l_7 = 0$:

$$\mathbf{N}(\mathbf{J}) \approx \begin{pmatrix} 0 \\ 0 \\ -0.73 \\ 0.69 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{G}^\top) \approx \begin{pmatrix} 0 \\ 0 \\ 0.51 \\ 0.86 \\ 0 \end{pmatrix}, \quad (28.63)$$

$$\mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ -0.42 \\ 0 \\ 0.57 \\ 0.42 \\ 0 \end{pmatrix}, \quad \mathbf{N}(\mathbf{J}^\top) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & -1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}. \quad (28.64)$$

Notice that changing either contact to SF makes it possible for the hand to resist external moments applied about the line containing the contacts, so the grasp loses indeterminacy, but retains graspability (with squeezing still possible along the line of the contacts). However, if contact 2 is the SF contact, the grasp loses its redundancy. While the second contact point can still be moved out of the plane by joint 3 and back in by joint 4, this canceled translation of the contact point yields a net rotation about $\hat{\mathbf{n}}_2$ (this also implies that the hand can control the moment applied to the object along the line containing the contacts). Changing to SF at contact 2 does *not* affect the hand's inability to move contact 1 and contact 2 in the $\hat{\mathbf{d}}_1$ and $\hat{\mathbf{n}}_2$ directions, so the defectivity property is retained.

Example 1, Part 5: Desirable Properties

Assuming contact model types of SF and HF at contacts 1 and 2, respectively, \mathbf{G} is full row rank and so $\mathcal{N}(\mathbf{G}^\top) = \mathbf{0}$ (Table 28.9). Therefore, as long as the hand is sufficiently dexterous, it can apply any wrench in \mathbb{R}^6 to the object. Also, if the joints are locked, object motion will be prevented. Assuming the same problem values used in the previous part of this problem, the matrix \mathbf{G}^\top is:

$$\mathbf{G}^\top = \begin{pmatrix} -c_1 & -s_1 & 0 & 0 & 0 & 0 \\ s_1 & -c_1 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & rs_1 & -rc_1 & 0 \\ 0 & 0 & 0 & -c_1 & -s_1 & 0 \\ -c_2 & -s_2 & 0 & 0 & 0 & 0 \\ s_2 & -c_2 & 0 & 0 & 0 & -r \\ 0 & 0 & 1 & rs_2 & -rc_2 & 0 \end{pmatrix}. \quad (28.65)$$

Bases for the three nontrivial null spaces are:

$$\mathbf{N}(\mathbf{J}^\top) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad (28.66)$$

$$\mathbf{N}(\mathbf{J}) \approx \begin{pmatrix} 0 \\ 0 \\ -0.73 \\ 0.69 \\ 0 \end{pmatrix},$$

$$N(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ -0.42 \\ 0 \\ 0 \\ 0.57 \\ 0.42 \\ 0 \end{pmatrix}. \quad (28.67)$$

Since $\mathcal{R}(\mathbf{J})$ is four dimensional and $\mathcal{N}(\mathbf{G})$ is one dimensional, the maximum dimension of $\mathcal{R}(\mathbf{J}) + \mathcal{N}(\mathbf{G})$ cannot be more than five, and therefore, the hand cannot control all possible object velocities, for example, the contact velocity $\mathbf{v}_{ec} = (0 \ 0 \ 0 \ 0.8 \ 0 \ 0)^T$ is in $\mathcal{N}(\mathbf{J}^T)$, and so cannot be controlled by the fingers. It is also equal to 0.6 times the third column of \mathbf{G}^T plus the fourth column of \mathbf{G}^T and therefore is in $\mathcal{R}(\mathbf{G}^T)$. Since the mapping between $\mathcal{R}(\mathbf{G})$ and $\mathcal{R}(\mathbf{G}^T)$ is one to one and onto, this uncontrollable contact velocity corresponds to a unique uncontrollable object velocity, $\mathbf{v} = (0 \ 0 \ 0.6 \ 1 \ 0 \ 0)$. In other words, the hand cannot cause the center of the sphere to translate in the z -direction, while also rotating about the x -axis (and not other axes simultaneously).

On the question of controlling all internal object forces, the answer is yes, since $\mathcal{N}(\mathbf{J}^T) \cap \mathcal{N}(\mathbf{G}) = \mathbf{0}$. This conclusion is clear from the fact that $\mathcal{N}(\mathbf{G})$ has nonzero values in the first, second, and sixth positions, while all columns of $\mathcal{N}(\mathbf{J}^T)$ have zeros in those positions.

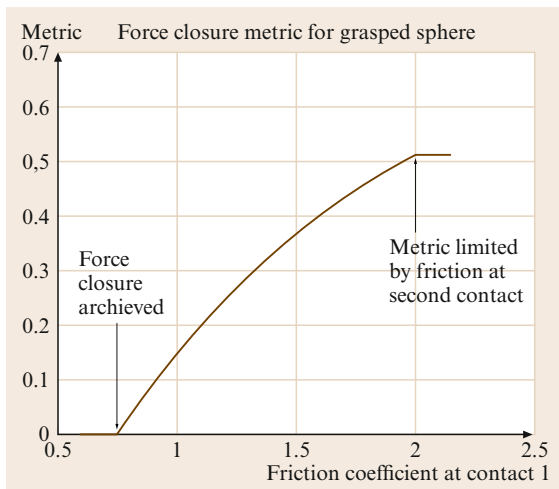


Fig. 28.13 Plot of the force closure metric versus the friction coefficient on contact 1

Example 1, Part6: Force Closure

Again assume that contacts 1 and 2 on the grasped sphere were modeled as SF and HF contacts, respectively. Under this assumption, \mathbf{G} is full row rank, and the internal force corresponds to equal and opposite contact forces. For frictional form closure to exist, the internal force must lie within the friction cones. Choosing r and the sines and cosines of θ_1 and θ_2 as in example 1, part 4, frictional form closure can be shown to exist if both friction coefficients are greater than 0.75. For this grasp, since $\mathcal{N}(\mathbf{J}^T) \cap \mathcal{N}(\mathbf{G}) = \mathbf{0}$, frictional form closure is equivalent to force closure.

The plot in Fig. 28.13 was generated by fixing $\mu_2 = 2.0$ and varying μ_1 . Notice that, for $\mu_1 < 0.75$, force closure does not exist. The metric increases smoothly until μ_1 becomes equal to μ_2 . At this point, increasing μ_1 further does not improve force closure, since contact 2 becomes the limiting factor. To increase the metric further, both friction coefficients must be increased.

28.5.2 Example 2: Grasped Polygon in the Plane

Part 1: \mathbf{G} and \mathbf{J}

Figure 28.14 shows a planar hand grasping a polygon. Finger 1 (on the right) contains two joints numbered 1 and 2. Finger 2 contains joints 3–7, which are numbered in increasing order moving from the palm distally. The inertial frame has been chosen to lie inside the object, with its x -axis passing through contacts 1 and 2, and collinear with the normal vector of contact 2.

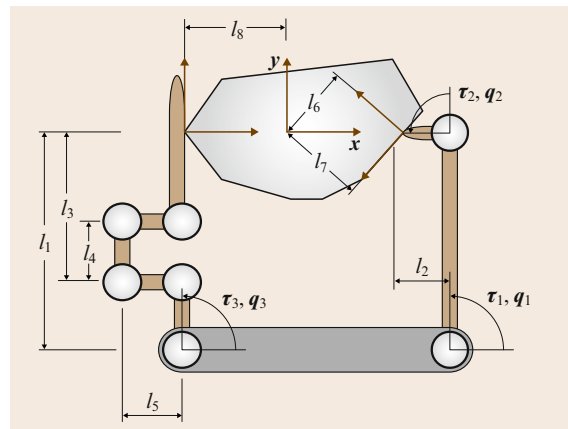


Fig. 28.14 Planar hand with two fingers and seven joints grasping a polygonal object

The rotation matrices are given by

$$\mathbf{R}_1 = \begin{pmatrix} -0.8 & -0.6 \\ 0.6 & -0.8 \end{pmatrix}, \quad \mathbf{R}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \quad (28.68)$$

Assuming HF contacts, \mathbf{G} is given as

$$\mathbf{G} = \begin{pmatrix} -0.8 & -0.6 & 1 & 0 \\ 0.6 & -0.8 & 0 & 1 \\ l_6 & -l_7 & 0 & -l_8 \end{pmatrix}. \quad (28.69)$$

Notice that the first two columns of \mathbf{G} correspond to the normal and tangential unit vectors at contact 1. The third and fourth columns correspond to contact 2.

Assuming HF contacts and that all joints are active (i. e., not locked), \mathbf{J} is

$$\mathbf{J}^\top = \begin{pmatrix} 0.8l_1 & 0.6l_1 & \mathbf{0} \\ -0.6l_2 & 0.8l_2 & \mathbf{0} \\ \mathbf{0} & -l_1 & 0 \\ & -l_3 & 0 \\ & -l_3 & l_5 \\ & -l_3 + l_4 & l_5 \\ & -l_3 + l_4 & 0 \end{pmatrix}. \quad (28.70)$$

The first two columns of \mathbf{J}^\top are the torques required to produce a unit force in the $\hat{\mathbf{n}}_1$ and $\hat{\mathbf{t}}_1$ directions at contact 1. The horizontal line through the matrix partitions the contributions for the first finger (the upper part) and second finger. Notice that both \mathbf{J}^\top and \mathbf{G} are full column rank.

Example 2, Part 2: Grasp Classes

This example clearly illustrates the physical qualities of the various grasp classes without introducing features that can cloud the descriptions.

We now discuss the details of the four grasp classes using the previous planar example. During these discussions it is useful to choose nondimensional values for the parameters in the grasping system. Assume l_4 is the unit length. The other lengths as fractions of l_4 are:

$$l_1 = 2.7, \quad l_2 = 1.0, \quad l_3 = 1.7, \quad (28.71)$$

$$l_4 = 1.0, \quad l_5 = 1.0, \quad l_6 = 1.0, \quad (28.72)$$

$$l_7 = 1.3, \quad l_8 = 1.5. \quad (28.73)$$

Redundant

Redundancy exists if $\mathcal{N}(\mathbf{J})$ is nontrivial. Assuming that both contacts are hard contacts and that all the joints

are active, $\text{rank}(\mathbf{J}) = 4$, so $\mathcal{N}(\mathbf{J})$ is three dimensional. A basis for $\mathcal{N}(\mathbf{J})$ was obtained as

$$\mathcal{N}(\mathbf{J}) \approx \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -0.50 & -0.24 & -0.18 \\ 0.53 & 0.67 & -0.10 \\ 0.48 & -0.49 & -0.02 \\ -0.48 & 0.49 & 0.02 \\ -0.02 & 0.01 & 0.98 \end{pmatrix}. \quad (28.74)$$

Since the first two rows are zero, $\mathcal{N}(\mathbf{J})$ does not include motions of the first finger (on the right of the palm). To understand this, assume that the object is fixed in the plane. Then the first finger cannot maintain sticking contact at contact 1 unless its joints are also fixed.

The three nonzero columns corresponding to finger 2 show that there are three basis motions of its joints that allow the finger contact to stick to the object contact. For example, the first column shows that, if joint 3 moves roughly as much as joints 4, 5, and 6, but in the opposite direction as joints 4 and 5 and in the same direction as joint 6, while joint 7 is more or less fixed, then contact 2 will be maintained.

Notice that finger 2 contains a parallelogram. Because of this geometry, one can see that the vector $(0 \ 0 \ 0 \ -1 \ 1 \ -1 \ 1)^\top$ is an element of $\mathcal{N}(\mathbf{J})$. The velocity interpretation of this vector is that the link of the finger connected to the palm, and the link touching the object remain fixed in space, while the parallelogram moves as a simple four-bar mechanism. Similarly, joint actions in $\mathcal{N}(\mathbf{J})$ do not affect the contact forces, but cause internal hand velocities. Also, notice that, since $\mathcal{N}(\mathbf{J}^\top) = \mathbf{0}$, the entire space of possible generalized velocities and forces at the contacts can be generated by the joints.

Indeterminate

As noted above, with HF contact models, the system is graspable. However, replacing the HF models with PwoF models removes the tangent force components in the $\hat{\mathbf{t}}_1$ and $\hat{\mathbf{t}}_2$ directions. This effectively removes columns 2 and 4 from \mathbf{G} , which guarantees that the system will be indeterminate. The reduced matrix is denoted by $\mathbf{G}_{(1,3)}$. In this case $\mathcal{N}(\mathbf{G}_{(1,3)}^\top)$ is

$$\mathcal{N}(\mathbf{G}_{(1,3)}^\top) \approx \begin{pmatrix} 0 \\ -0.86 \\ 0.51 \end{pmatrix}. \quad (28.75)$$

Physically, this basis vector corresponds to moving the object such that the point coincident with the origin of $\{N\}$ moves directly downward, while the object rotates counterclockwise. Also, if the analogous force and moment were applied to the object, the frictionless contacts could not maintain equilibrium.

Grasable

With two HF contact models in force, $\text{rank}(\mathbf{G}) = 3$, so $\mathcal{N}(\mathbf{G})$ is one dimensional and the system is grasable. The null space basis vector of the grasp matrix is:

$$\mathbf{N}(\mathbf{G}) \approx \begin{pmatrix} 0.57 \\ 0.42 \\ \frac{0.71}{0} \end{pmatrix}. \quad (28.76)$$

The physical interpretation of this basis vector is two opposing forces acting through the two contact points. Recall that, because the contact model is kinematic, there is no consideration of contact friction. However, given the direction of the contact normal relative to the line of the internal force, one can see that if the coefficient of friction is not greater than 0.75, squeezing tightly will cause sliding at contact 1, thus violating the kinematic contact model.

Defective

In a defective grasp, $\mathcal{N}(\mathbf{J}^\top) \neq \mathbf{0}$. Given that the original \mathbf{J} is full row rank, the grasp is *not* defective. However, it can be made defective by locking a number of joints and/or changing the hand's configuration so that \mathbf{J} is no longer full rank. For example, locking joints 4, 5, 6, and 7 makes finger 2 a single-link finger with only joint 3 active. In this new grasping system, $\mathbf{J}_{(1,2,3)}^\top$ is simply the first three rows of the original \mathbf{J}^\top given in (28.70), where the subscript is the list of indices of active joints. The null space basis vector is:

$$\mathbf{N}(\mathbf{J}_{(1,2,3)}^\top) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \quad (28.77)$$

This grasp is defective, since there is a subspace of contact velocities and forces that cannot be controlled by joint generalized velocities and forces. Since only the last component of $\mathbf{N}(\mathbf{J}_{(1,2,3)}^\top)$ is nonzero, it

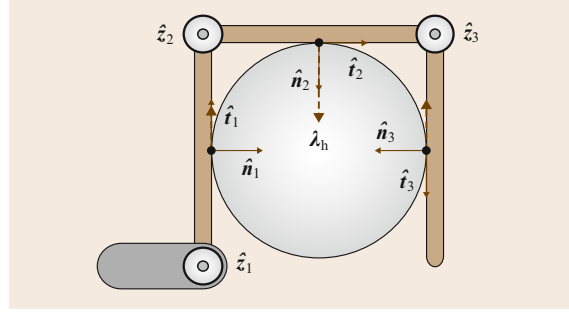


Fig. 28.15 A sphere grasped by a finger with three revolute joints. The force direction λ_h (dashed line) is a force that belongs to both $\mathcal{N}(\mathbf{G})$ and $\mathcal{N}(\mathbf{J}^\top)$ and causes hyperstaticity

would be impossible for the hand to give the contact point 2 on the object a velocity in the \hat{t}_2 -direction while maintaining the contact. This is also clear from the arrangement of joint 3, contact 2, and the direction of the contact normal. The dual interpretation is that forces in $\mathcal{N}(\mathbf{J}^\top)$ are resisted by the structure and the corresponding joint loads is zero, or equivalently that those forces are not controllable by the hand. Notice that if the model of contact 2 were changed to point without friction, then $\mathbf{N}(\mathbf{J}_{(1,2,3)}^\top) = \mathbf{0}$ and the system would no longer be defective.

28.5.3 Example 3: Hyperstatic Grasps

Part 1: $\tilde{\mathbf{G}}$ and $\tilde{\mathbf{J}}$

Figure 28.15 shows a planar projection of a three-dimensional sphere of radius l grasped by one finger only, with three revolute joints, through three contacts. The frames $\{C\}_1$, $\{C\}_2$, and $\{C\}_3$ are oriented so that their \hat{o} -directions point out of the plane of the figure (as indicated by the small bold circle). The axes of the frames $\{N\}$ and $\{B\}$ were chosen to be axis-aligned with coincident origins located at the center of the sphere. The z -axes are pointing out of the page. Observe that since the three joint axes of the finger are perpendicular to the (x, y) -plane, the grasp operates in that plane for all time.

Assume that the width of all the links of the robotic hand is zero. Rotation matrices \mathbf{R}_i and vectors $\mathbf{c}_i - \mathbf{p}$ for $i = 1, \dots, 3$, can be computed as in (28.48) and (28.49) considering that $\theta_1 = \pi$, for contact 1, $\theta_2 = \pi/2$ and $\theta_3 = 0$, for contacts 2 and 3, respectively. Finally, the complete grasp matrix is $\tilde{\mathbf{G}}^\top = (\tilde{\mathbf{G}}_1 \ \tilde{\mathbf{G}}_2 \ \tilde{\mathbf{G}}_3)^\top \in \mathbb{R}^{18 \times 6}$

where $\tilde{\mathbf{G}}_i$ is as defined in (28.50):

$$\tilde{\mathbf{G}}^\top = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & l & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & l & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & -l & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Construction of the complete hand Jacobian $\tilde{\mathbf{J}}_i$ for contact i requires knowledge of the joint axis directions and the origins of the frames fixed to the links of each finger. Assume that the origins of the DH frames lie in the plane of the figure. In the current configuration, the quantities of interest for contact 1, expressed directly in $\{N\}$, are

$$\begin{aligned} \mathbf{c}_1 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} 0 & l & 0 \end{pmatrix}^\top, \\ \hat{\mathbf{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top. \end{aligned}$$

The quantities of interest for contact 2, in $\{N\}$, are

$$\begin{aligned} \mathbf{c}_2 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} l & 2l & 0 \end{pmatrix}^\top, \\ \mathbf{c}_2 - \boldsymbol{\zeta}_2 &= \begin{pmatrix} l & 0 & 0 \end{pmatrix}^\top, \\ \hat{\mathbf{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top, \\ \hat{\mathbf{z}}_2 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top. \end{aligned}$$

The quantities of interest for contact 3, in $\{N\}$, are

$$\begin{aligned} \mathbf{c}_3 - \boldsymbol{\zeta}_1 &= \begin{pmatrix} 2l & l & 0 \end{pmatrix}^\top, \\ \mathbf{c}_3 - \boldsymbol{\zeta}_2 &= \begin{pmatrix} 2l & -l & 0 \end{pmatrix}^\top, \end{aligned}$$

$$\begin{aligned} \mathbf{c}_3 - \boldsymbol{\zeta}_3 &= \begin{pmatrix} 0 & -l & 0 \end{pmatrix}^\top, \\ \hat{\mathbf{z}}_1 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top, \\ \hat{\mathbf{z}}_2 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top, \\ \hat{\mathbf{z}}_3 &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}^\top. \end{aligned}$$

The complete hand Jacobian $\tilde{\mathbf{J}} \in \mathbb{R}^{18 \times 3}$ (contact velocities are expressed in $\{\mathbf{C}\}_i$) is:

$$\tilde{\mathbf{J}} = \begin{pmatrix} -l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ \hline -l & -l & 0 \\ -2l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline l & -l & -l \\ -2l & -2l & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}.$$

The horizontal dividing line partitions $\tilde{\mathbf{J}}$ into $\tilde{\mathbf{J}}_1$ (on top), $\tilde{\mathbf{J}}_2$, and $\tilde{\mathbf{J}}_3$ (on the bottom). The columns correspond to joints 1–3.

Example 3, Part 2: \mathbf{G} and \mathbf{J}

Assume that the three contacts in Fig. 28.15 are of type HF. Then the selection matrix \mathbf{H} is given by

$$\mathbf{H} = \begin{pmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{pmatrix}, \quad (28.78)$$

where \mathbf{I} and $\mathbf{0}$ are in $\mathbb{R}^{3 \times 3}$, thus the matrices $\mathbf{G}^\top \in \mathbb{R}^{9 \times 6}$ and $\mathbf{J} \in \mathbb{R}^{9 \times 3}$ are obtained by removing rows related to

rotations from $\tilde{\mathbf{G}}^\top$ and $\tilde{\mathbf{J}}$:

$$\mathbf{G}^\top = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & l & 0 \\ \hline 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & l & 0 & 0 \\ \hline -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & -l \\ 0 & 0 & 1 & 0 & -l & 0 \end{pmatrix},$$

$$\mathbf{J} = \begin{pmatrix} l & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline l & l & 0 \\ 2l & 0 & 0 \\ 0 & 0 & 0 \\ \hline l & l & l \\ 2l & 2l & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

Example 3, Part 3: Grasp Classes

The first column of Table 28.10 reports the dimensions of the main subspaces of \mathbf{J}^\top and \mathbf{G} for the sphere grasping example with three hard-finger contacts. Only nontrivial null spaces are listed.

The system is defective because there are generalized contact forces belonging to the subspace that are resisted by the structure, which correspond to zero joint actions:

$$\mathbf{N}(\mathbf{J}^\top) = \begin{pmatrix} 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

The first three columns represent generalized forces acting at the three contact points in a direction perpendicular to the plane of the Fig. 28.15. The fourth column corresponds to a contact force applied only along the $\hat{\mathbf{t}}_1$ direction.

Table 28.10 Dimensions of the main subspaces and classifications of the grasp given in Example 3

Dimension	Class
$\dim \mathcal{N}(\mathbf{J}^\top) = 6$	Defective
$\dim \mathcal{N}(\mathbf{G}) = 3$	Graspable
$\dim \mathcal{N}(\mathbf{J}^\top) \cap \mathcal{N}(\mathbf{G}) = 1$	Hyperstatic

The system is graspable because the subspace of internal forces is three dimensional; a possible basis matrix is:

$$\mathbf{N}(\mathbf{G}) = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \\ \hline 1 & 0 & 2 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{pmatrix}.$$

The three force vectors of the subspace $\mathbf{N}(\mathbf{G})$ are easily identified from Fig. 28.15. Note that all forces are expressed in local contact frames. The first column vector of $\mathbf{N}(\mathbf{G})$ represents opposed forces at contacts 1 and 2 along the line joining contacts 1 and 2. The second column vector parameterizes opposed forces at contacts 1 and 3 along the line joining contacts 1 and 3. The last vector represents forces along direction λ_h , shown as the dashed lines in Fig. 28.15. Note that this direction (in wrench intensity space) corresponds to two upward friction forces at the left and right contacts and one downward with double the magnitude from the center of the top link in the work space.

Finally, the grasp is hyperstatic because

$$\mathbf{N}(\mathbf{G}) \cap \mathbf{N}(\mathbf{J}^\top) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ \hline 2 \\ 0 \\ 0 \\ \hline 0 \\ 0 \\ -1 \\ 0 \end{pmatrix} \neq 0.$$

Hyperstatic forces in this subspace are internal forces that cannot be controlled through the hand joints. In

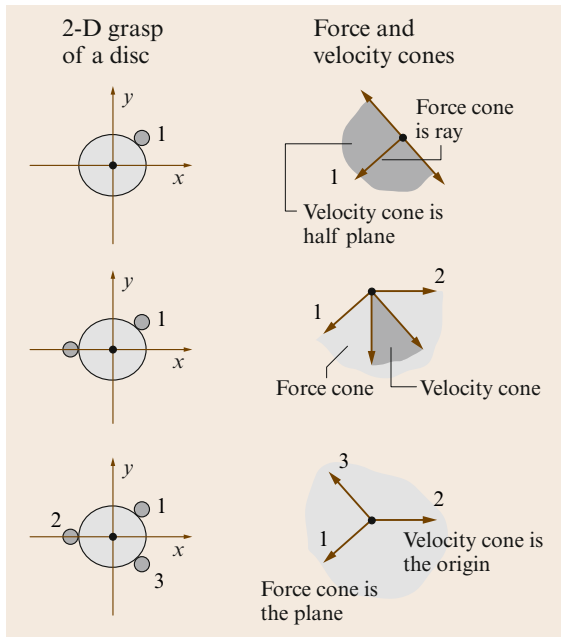


Fig. 28.16 The case of a translating disc in the plane: relationship between frictionless contacts and possible disc velocities and net contact forces

Fig. 28.15 the internal force λ_h that is also in $N(J^\top)$ is reported.

The grasp in Fig. 28.15 is an example of a *power grasp*, a style of grasp mentioned earlier that uses many contact points not only on the fingertips but also on the links of fingers and the palm [28.4, 20, 31].

All power grasps are kinematically defective ($N(J^\top) \neq \mathbf{0}$) and are usually hyperstatic. According to

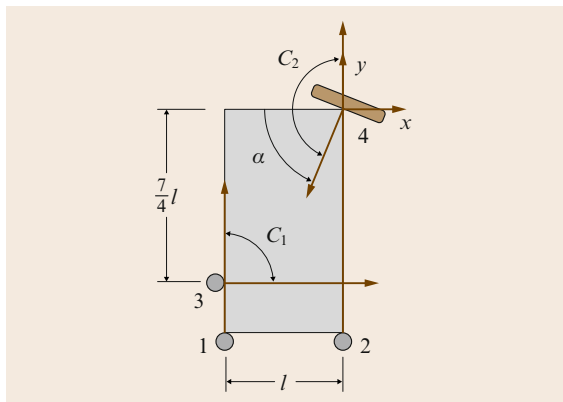


Fig. 28.17 Planar grasp with first-order form closure if $1.052 < \alpha < \frac{\pi}{2}$

Sect. 28.4.2, rigid-body modeling is not sufficient to capture the overall system behavior, because the generalized contact forces in $N(G) \cap N(J^\top)$ leave the dynamics indeterminate.

Many approaches have been used to overcome the rigid-body limitation in hyperstatic grasps such as those proposed in [28.12, 16, 17] where viscoelastic contact models have been used to solve the force indeterminacy. In [28.32], the authors found that a sufficient condition for hyperstaticity is $m > q + 6$, where m is the dimension of the contact force vector.

28.5.4 Example 4: Duality

Consider a frictionless disc constrained to translate in the plane (Fig. 28.16). In this problem $n_v = 2$, so the space of applied contact forces and object velocities is the plane \mathbb{R}^2 . In the top pair of pictures, a single (fixed) contact point imposes a half-space constraint on the instantaneous velocity and limits the force at a frictionless contact to the ray. Both the ray and the (dark gray) half space are defined by the contact normal pointing into the object. Notice that the ray and half space are dual cones. When two contacts are present, the (light gray) force cone becomes the nonnegative span of the two contact normals and the velocity cone is its dual. With the addition of the third contact, the grasp has form closure, as indicated by the degeneration of the velocity cone to the origin and the expansion of the force cone to become equal to the plane.

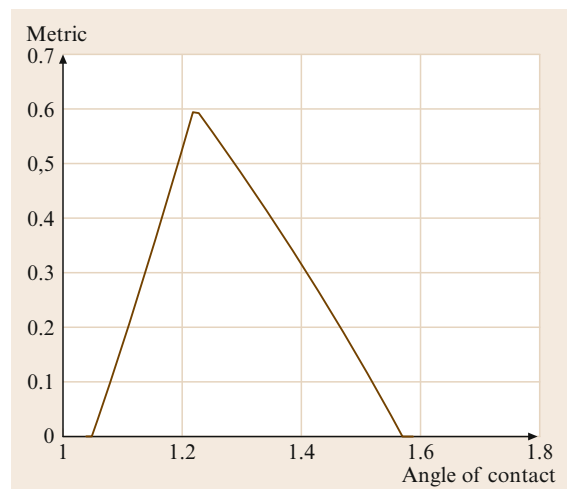


Fig. 28.18 Plot of closure metric versus angle of contact if $1.04 < \alpha < 1.59$

It is important to point out that the discussion of the dual cones applies to three-dimensional bodies after replacing the contact normals with the columns of G .

28.5.5 Example 5: Form Closure

Form closure of a spatial object requires seven contacts, which is difficult to illustrate. Therefore, the only form closure example analyzed in this chapter is for the following planar problem.

The fundamental features of a planar grasp with four contacts are shown in Fig. 28.17. Notice that the normal at the fourth contact is well defined despite the contact occurring on the vertex of the object. The angle α of the finger is allowed to vary, and it can be shown that form

closure exists if α lies in the interval $1.0518 < \alpha < \frac{\pi}{2}$. Notice that a critical value of α occurs when the lower edge of \mathcal{C}_2 contains contact point 3 ($\alpha \approx 1.0518$) and contact point 2 ($\alpha = \frac{\pi}{2}$).

Choosing the frame for analysis with origin at the fourth contact point, the grasp matrix for this example is

$$G = \begin{pmatrix} 0 & 0 & 1 & -\cos(\alpha) \\ 1 & 1 & 0 & -\sin(\alpha) \\ -l & 0 & \frac{7}{4}l & 0 \end{pmatrix}. \quad (28.79)$$

Form closure was tested for a range of angles as shown in the plot in Fig. 28.18, which indicates that the grasp farthest from losing form closure has $\alpha \approx 1.22$ radians, which is the configuration shown in the figure.

28.6 Conclusion and Further Reading

A great deal of understanding of grasping systems can be derived from the simple linear kinematic, dynamic, and contact models. The most widely used grasp classifications and closure properties can all be derived from these models under the rigid-body assumption. Linearizing these models leads to metrics and tests that can be computed efficiently using computational linear algebra and linear programming techniques. In-depth discussions of grasp kinematics and grasp classification can be found in [28.11, 16, 21, 32–36]. When making simplifying assumptions, one has to wonder what insights have been lost. For the interested reader, there are a host of papers that analyze grasping system under more sophisticated assumptions. In general, bodies are curved and compliant [28.13, 22, 31, 37–39]. Contact friction models are not quite as simple as the Coulomb approximation so widely adopted, and presented in this chapter; for example, if a contact has to resist a moment about its normal, its effective tangential friction coefficient is reduced [28.29]. In our analysis, the quadratic Coulomb friction cone was approximated by a polyhedral cone. The analysis problems are more difficult when using the quadratic cone, but they are quite tractable [28.30, 40]. In principle, a properly designed hand–object system could be controlled to maintain all contacts, but worldly realities can lead to unwanted slipping or twisting.

Other important research topics within the area of grasping are: grasp synthesis, force distribution, stability, and dexterous manipulation. Grasp synthesis is the problem of choosing the posture of the hand and contact point locations to optimize a grasp quality metric. Note that this is quite a different issue from *grasp*

acquisition, which refers to the act of achieving the chosen grasp. Grasp acquisition is an act of dexterous manipulation. One of the first studies of grasp synthesis for multifingered hands was undertaken by *Jameson* in 1985 [28.41] who designed a Levenberg–Marquardt algorithm to search the surface of an object for the locations of three points that would achieve force closure. Since this work, many other metrics and approaches to searching for high-quality grasps have been implemented (see, for example, [28.42–47]). A recent paper by *Liu et al.* provides an in-depth discussion of several quality metrics. They further demonstrate that, if one chooses finger contact locations by optimizing the metrics and if a force closure grasp exists, then the grasps obtained are force closed [28.48]. For grasping of several objects simultaneously, see [28.49].

The grasp force distribution problem is that of finding *good* joint actions and contact forces to balance a given external load applied to the object. This problem was studied in the context of walking machines first by *McGhee* and *Orin* [28.50] and later by several others [28.51, 52]. Kumar and Waldron applied similar techniques to force distribution problems in grasping [28.53]. Recent work by *Han et al.* and *Buss et al.* has solved the force distribution problem with nonlinear friction cone constraints by taking advantage of second-order cone programming and related techniques [28.30, 40, 54]. In power grasps, this problem is more involved because contact forces that cannot be applied may exist, as studied in [28.17, 19, 20, 55].

Grasp stability is too often equated to grasp closure, because all external forces can be balanced by the hand.

However, grasp closure is really equivalent to the existence of equilibrium, which is a necessary condition for stability, but not a sufficient condition. The common definition of stability outside of the field of robotic grasping requires that, when a system is deflected from an equilibrium point, the system returns to this point. This was the approach taken by *Hanafusa* and *Asada*, who studied grasp stability for a hand with three elastic fingers [28.56]. *Cutkosky* took the analysis to three dimensions and accounted for finger and object curvatures [28.37]. He showed that stability was dependent on the stiffnesses of the fingers and controller and on the curvatures of the bodies in contact.

Dexterous manipulation research first appeared in the literature in the 1970s. *Hanafusa* and *Asada* developed a plan to turn a nut onto a bolt [28.57]. Since then a progression of increasingly complex manipulation tasks have been studied at varying degrees of detail. *Mason* and many of his students studied pushing tasks in the horizontal plane, see for example [28.58–61]. Other notable planar work can be found in [28.62–66]. The planning and execution of dexterous manipulation tasks in three dimensions continues to be an open problem in robotics. However, for several interesting approaches and many useful insights, the following reading is recommended [28.67–72].

References

- 28.1 J.K. Salisbury: Kinematic and Force Analysis of Articulated Hands. Ph.D. Thesis (Stanford University, Stanford 1982)
- 28.2 A.T. Miller, P.K. Allen: GraspIt! A versatile simulator for robotic grasping, *IEEE Robot. Autom. Mag.* **11**(4), 110–122 (2004)
- 28.3 A. Bicchi: Hands for dextrous manipulation and powerful grasping: A difficult road towards simplicity, *IEEE Trans. Robot. Autom.* **16**, 652–662 (2000)
- 28.4 K. Salisbury, W. Townsend, B. Ebrman, D. DiPietro: Preliminary design of a whole-arm manipulation system (WAMS), *Proc. IEEE Int. Conf. Robot. Autom.* (1988) pp. 254–260
- 28.5 M.S. Ohwovoriole, B. Roth: An extension of screw theory, *J. Mech. Des.* **103**, 725–735 (1981)
- 28.6 K.H. Hunt: *Kinematic Geometry of Mechanisms* (Oxford Univ. Press, Oxford 1978)
- 28.7 T.R. Kane, D.A. Levinson, P.W. Likins: *Spacecraft Dynamics* (McGraw Hill, New York 1980)
- 28.8 J.J. Craig: *Introduction to Robotics: Mechanics and Control*, 2nd edn. (Addison-Wesley, Reading 1989)
- 28.9 J.K. Salisbury, B. Roth: Kinematic and force analysis of articulated mechanical hands, *J. Mech. Trans. Autom. Des.* **105**, 35–41 (1983)
- 28.10 M.T. Mason, J.K. Salisbury Jr: *Robot Hands and the Mechanics of Manipulation* (MIT Press, Cambridge 1985)
- 28.11 A. Bicchi: On the closure properties of robotic grasping, *Int. J. Robot. Res.* **14**(4), 319–334 (1995)
- 28.12 D. Prattichizzo, A. Bicchi: Consistent task specification for manipulation systems with general kinematics, *ASME J. Dyn. Syst. Meas. Contr.* **119**, 760–767 (1997)
- 28.13 J. Kerr, B. Roth: Analysis of multifingered hands, *Int. J. Robot. Res.* **4**(4), 3–17 (1986)
- 28.14 G. Strang: *Introduction to Linear Algebra* (Wellesley-Cambridge, Wellesley 1993)
- 28.15 R.M. Murray, Z. Li, S.S. Sastry: A Mathematical Introduction to Robot Manipulation (CRC Press, Boca Raton 1993)
- 28.16 D. Prattichizzo, A. Bicchi: Dynamic analysis of mobility and graspability of general manipulation systems, *IEEE Trans. Robot. Autom.* **14**(2), 241–258 (1998)
- 28.17 A. Bicchi: On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation, *Int. J. Robot. Auton. Syst.* **13**, 127–147 (1994)
- 28.18 F. Reuleaux: *The Kinematics of Machinery* (Macmillan, New York 1876), Republished by Dover, New York (1963)
- 28.19 T. Omata, K. Nagata: Rigid body analysis of the indeterminate grasp force in power grasps, *IEEE Trans. Robot. Autom.* **16**(1), 46–54 (2000)
- 28.20 J.C. Trinkle: The Mechanics and Planning of Enveloping Grasps. Ph.D. Thesis (University of Pennsylvania, Philadelphia 1987)
- 28.21 K. Lakshminarayana: Mechanics of form closure, *Amer. Soc. Mech. Eng. Tech. Rep.* **78-DET-32** (1978)
- 28.22 E. Rimon, J. Burdick: Mobility of bodies in contact i: A 2nd order mobility index for multiple-finger grasps, *IEEE Trans. Robot. Autom.* **14**(5), 696–708 (1998)
- 28.23 B. Mishra, J.T. Schwartz, M. Sharir: On the existence and synthesis of multifinger positive grips, *Algorithmica* **2**(4), 541–558 (1987)
- 28.24 P. Somov: Über Schraubengeschwindigkeiten eines festen Körpers bei verschiedener Zahl von Stützflächen, *Z. Math. Phys.* **42**, 133–153 (1897), (in German)
- 28.25 A.J. Goldman, A.W. Tucker: Polyhedral convex cones. In: *Linear Inequalities and Related Systems*, ed. by H.W. Kuhn, A.W. Tucker (Princeton Univ., York 1956) pp. 19–40

- 28.26 X. Markenscoff, L. Ni, C.H. Papadimitriou: The geometry of grasping, *Int. J. Robot. Res.* **9**(1), 61–74 (1990)
- 28.27 D.G. Luenberger: *Linear and Nonlinear Programming*, 2nd edn. (Addison-Wesley, Reading 1984)
- 28.28 V.D. Nguyen: The synthesis of force closure grasps in the plane. M.S. Thesis (MIT, Cambridge 1985), AI-TR861
- 28.29 R.D. Howe, M.R. Cutkosky: Practical force-motion models for sliding manipulation, *Int. J. Robot. Res.* **15**(6), 557–572 (1996)
- 28.30 L. Han, J.C. Trinkle, Z. Li: Grasp analysis as linear matrix inequality problems, *IEEE Trans. Robot. Autom.* **16**(6), 663–674 (2000)
- 28.31 M.R. Cutkosky, I. Kao: Computing and controlling the compliance of a robotic hand, *IEEE Trans. Robot. Autom.* **5**(2), 151–165 (1989)
- 28.32 J.C. Trinkle: On the stability and instantaneous velocity of grasped frictionless objects, *IEEE Trans. Robot. Autom.* **8**(5), 560–572 (1992)
- 28.33 K.H. Hunt, A.E. Samuel, P.R. McAree: Special configurations of multi-finger multi-freedom grippers – a kinematic study, *Int. J. Robot. Res.* **10**(2), 123–134 (1991)
- 28.34 D.J. Montana: The kinematics of multi-fingered manipulation, *IEEE Trans. Robot. Autom.* **11**(4), 491–503 (1995)
- 28.35 Y. Nakamura, K. Nagai, T. Yoshikawa: Passive and active closures by constraining mechanisms, *Int. J. Robot. Res.* **8**, 44–61 (1989)
- 28.36 J.S. Pang, J.C. Trinkle: Stability characterizations of rigid body contact problems with coulomb friction, *Z. Angew. Math. Mech.* **80**(10), 643–663 (2000)
- 28.37 M.R. Cutkosky: *Robotic Grasping and Fine Manipulation* (Kluwer Academic, Norwell 1985)
- 28.38 W.S. Howard, V. Kumar: On the stability of grasped objects, *IEEE Trans. Robot. Autom.* **12**(6), 904–917 (1996)
- 28.39 A.B.A. Cole, J.E. Hauser, S.S. Sastry: Kinematics and control of multifingered hands with rolling contacts, *IEEE Trans. Autom. Contr.* **34**, 398–404 (1989)
- 28.40 M. Buss, H. Hashimoto, J. Moore: Dexterous hand grasping force optimization, *IEEE Trans. Robot. Autom.* **12**(3), 406–418 (1996)
- 28.41 J. Jameson: Analytic Techniques for Automated Grasp. Ph.D. Thesis (Stanford University, Stanford 1985)
- 28.42 V. Nguyen: Constructing force-closure grasps, *Int. J. Robot. Res.* **7**(3), 3–16 (1988)
- 28.43 N.S. Pollard: Parallel algorithms for synthesis of whole-hand grasps, *Proc. IEEE Int. Conf. Robot. Autom.* (1997)
- 28.44 Y.C. Park, G.P. Starr: Grasp synthesis of polygonal objects using a three-fingered robot hand, *Int. J. Robot. Res.* **11**(3), 163–184 (1992)
- 28.45 I.M. Chen, J.W. Burdick: Finding antipodal point grasps on irregularly shaped objects, *IEEE Trans. Robot. Autom.* **9**(4), 507–512 (1993)
- 28.46 B. Mishra: Grasp metrics: Optimality and complexity, *Proc. Workshop Algorithmic Found. Robot.* (Peters, Boston 1994)
- 28.47 J. Ponce, S. Sullivan, A. Sudsang, J.-D. Boissonnat, J.-P. Merlet: On computing four-finger equilibrium and force-closure grasps of polyhedral objects, *Int. J. Robot. Res.* **16**(1), 11–35 (1997)
- 28.48 G.F. Liu, J. Xu, X. Wang, Z.X. Li: On quality functions for grasp synthesis, fixture planning, and coordinated manipulation, *IEEE Trans. Autom. Sci. Eng.* **1**(2), 146–162 (2004)
- 28.49 K. Harada, M. Kaneko: Neighborhood equilibrium grasp for multiple objects, *Proc. IEEE Int. Conf. Robot. Autom.* (2000) pp. 2159–2164
- 28.50 R.B. McGhee, D.E. Orin: A mathematical programming approach to control of positions and torques in legged locomotion systems, *Proc. RO-MANCY* (1976)
- 28.51 K. Waldron: Force and motion management in legged locomotion, *IEEE J. Robot. Autom.* **2**(4), 214–220 (1986)
- 28.52 T. Yoshikawa, K. Nagai: Manipulating and grasping forces in manipulation by multi-fingered grippers, *Proc. IEEE Int. Conf. Robot. Autom.* (1987) pp. 1998–2007
- 28.53 V. Kumar, K. Waldron: Force distribution in closed kinematic chains, *IEEE J. Robot. Autom.* **4**(6), 657–664 (1988)
- 28.54 M. Buss, L. Faybusovich, J. Moore: Dikin-type algorithms for dexterous grasping force optimization, *Int. J. Robot. Res.* **17**(8), 831–839 (1998)
- 28.55 D. Prattichizzo, J.K. Salisbury, A. Bicchi: Contact and grasp robustness measures: Analysis and experiments. In: *Experimental Robotics IV*, Lecture Notes Contr. Inf. Sci., Vol. 223, ed. by O. Khatib, K. Salisbury (Springer, Berlin, Heidelberg 1997)
- 28.56 H. Hanafusa, H. Asada: Stable prehension by a robot hand with elastic fingers. In: *Robot Motion: Planning and Control*, ed. by M. Brady, J. Hollerbach, T. Johnson, T. Lozano-Perez, M. Mason (MIT Press, Cambridge 1982) pp. 323–336
- 28.57 H. Hanafusa, H. Asada: Handling of constrained objects by active elastic fingers and its applications to assembly, *Trans. Soc. Instrum. Contr. Eng.* **15**(1), 61–66 (1979)
- 28.58 M.T. Mason: Manipulator Grasping and Pushing Operations. Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1982), reprinted in *Robot Hands and the Mechanics of Manipulation* MIT Press, Cambridge 1985
- 28.59 R.C. Brost: Analysis and Planning of Planar Manipulation Tasks. Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1991)
- 28.60 M.A. Peshkin, A.C. Sanderson: Planning robotic manipulation strategies for workpieces that slide, *IEEE J. Robot. Autom.* **4**(5), 524–531 (1988)

- 28.61 K. Lynch: Nonprehensile Manipulation: Mechanics and Planning. Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1996)
- 28.62 J.C. Trinkle, J.J. Hunter: A framework for planning dexterous manipulation, Proc. IEEE Int. Conf. Robot. Autom. (1991) pp. 1245–1251
- 28.63 J.C. Trinkle, R.C. Ram, A.O. Farahat, P.F. Stiller: Dexterous manipulation planning and execution of an enveloped slippery workpiece, Proc. IEEE Int. Conf. Robot. Autom., Vol. 2 (1993) pp. 442–448
- 28.64 K. Harada, M. Kaneko, T. Tsuji: Rolling based manipulation for multiple objects, Proc. IEEE Int. Conf. Robot. Autom. (2000) pp. 3887–3894
- 28.65 R.S. Fearing: Simplified grasping and manipulation with dextrous robot hands, IEEE J. Robot. Autom. **RA-2**(4), 188–195 (1986)
- 28.66 N.B. Zumel, M.A. Erdmann: Nonprehensile two palm manipulation with non-equilibrium transitions between stable states, Proc. IEEE Int. Conf. Robot. Autom. (1996) pp. 3317–3323
- 28.67 D.L. Brock: Enhancing the dexterity of a robot hand using controlled slip. M.S. Thesis (MIT, Cambridge 1987)
- 28.68 R. Fearing, S. Gopalswamy: Grasping polyhedral objects with slip, Proc. IEEE Int. Conf. Robot. Autom. (1989) pp. 296–301
- 28.69 M. Cherif, K.K. Gupta: Planning quasi-static fingertip manipulation for reconfiguring objects, IEEE Trans. Robot. Autom. **15**(5), 837–848 (1999)
- 28.70 L. Han, J.C. Trinkle: Dextrous manipulation by rolling and finger gaiting, Proc. IEEE Int. Conf. Robot. Autom. (1998) pp. 730–735
- 28.71 L. Han, Z. Li, J.C. Trinkle, Z. Qin, S. Jiang: The planning and control of robot dexterous manipulation, Proc. IEEE Int. Conf. Robot. Autom. (2000) pp. 263–269
- 28.72 M. Higashimori, M. Kimura, I. Ishii, M. Kaneko: Friction independent dynamic capturing strategy for a 2d stick-shaped object, Proc. IEEE Int. Conf. Robot. Autom. (2007) pp. 217–224

Cooperative

29. Cooperative Manipulators

Fabrizio Caccavale, Masaru Uchiyama

This chapter is devoted to cooperative manipulation of a common object by means of two or more robotic arms. The chapter opens with a historical overview of the research on cooperative manipulation, ranging from early 1970s to very recent years. Kinematics and dynamics of robotic arms cooperatively manipulating a tightly grasped rigid object are presented in depth. As for the kinematics and statics, the chosen approach is based on the so-called symmetric formulation; fundamentals of dynamics and reduced-order models for closed kinematic chains are discussed as well. A few special topics, such as the definition of geometrically meaningful cooperative task space variables, the problem of load distribution, and the definition of manipulability ellipsoids, are included to give the reader a complete picture of modeling and evaluation methodologies for cooperative manipulators. Then, the chapter presents the main strategies for controlling both the motion of the cooperative system and the interaction forces between the manipulators and the grasped object; in detail, fundamentals of hybrid force/position control, proportional-derivative (PD)-type force/position schemes, feedback linearization techniques, and impedance control approaches are given. In the last section further reading on advanced topics

29.1	A Historical Overview	701
29.2	Kinematics and Statics	703
29.2.1	Symmetric Formulation	704
29.2.2	Multifingered Manipulation	706
29.3	Cooperative Task Space	707
29.4	Dynamics and Load Distribution	708
29.4.1	Reduced-Order Models	708
29.4.2	Load Distribution	709
29.5	Task-Space Analysis	710
29.6	Control	711
29.6.1	Hybrid Control	711
29.6.2	PD Force/Motion Control	712
29.6.3	Feedback Linearization Approaches	713
29.6.4	Impedance Control	713
29.7	Conclusions and Further Reading	715
	References	716

related to control of cooperative robots is suggested; in detail, advanced nonlinear control strategies are briefly discussed (i.e., intelligent control approaches, synchronization control, decentralized control); also, fundamental results on modeling and control of cooperative systems possessing some degree of flexibility are briefly outlined.

29.1 A Historical Overview

It was not long after the emergence of robotics technologies that multi-arm robot systems began to be investigated by robotics researchers. In the early 1970s, research on this topic had already started. This interest was mainly due to typical limitations in applications of single-arm robots. It has been recognized, in fact, that many tasks that are difficult or impossible to execute by a single robot become feasible when two

or more manipulators are employed in a cooperative way. Such tasks include, for instance, carrying heavy or large payloads, the assembly of multiple parts without using special fixtures, and handling of objects that are flexible or possess extra degrees of freedom. Research on this subject has been aimed at solving existing problems and opening up a new stream of applications in flexible manufacturing systems as well as in

poorly structured environments (e.g., outer space and undersea).

Examples of research work in the early days include that by *Fujii* and *Kurono* [29.1], *Nakano* et al. [29.2], and *Takase* et al. [29.3]. Already in those pieces of work important key issues in the control of multi-arm robots were investigated: master/slave control, force/compliance control, and task-space control. In [29.1] *Fujii* and *Kurono* proposed to adopt a compliance control concept for the coordination of multiple manipulators; they defined a task vector with respect to the object frame and controlled the compliance expressed in the same coordinate frame. An interesting feature of the work by *Fujii* and *Kurono* [29.1] and by *Takase* et al. [29.3] is that force/compliance control was implemented by exploiting back-drivability of the actuators, without using force/torque sensors. The importance of this technique in practical applications, however, was not recognized at that time, and more complex approaches using force/torque sensors lured people in robotics. *Nakano* et al. [29.2, 4] proposed a master/slave force control approach for the coordination of two arms carrying an object cooperatively and pointed out the necessity of force control for cooperative multiple robots.

In the 1980s, based on several fundamental theoretical results for single-arm robots, strong research on multi-arm robotic systems was renewed [29.5]. Definition of task vectors with respect to the object to be handled [29.6], dynamics and control of the closed kinematic chain formed by the multi-arm robot and the object [29.7, 8], and force control issues, such as hybrid position/force control [29.9–12], were explored. Through this research work, a strong theoretical background for the control of multi-arm robots has been formed, providing the basis for research on more advanced topics from the 1990s to today.

How to parameterize the constraint forces/moments on the object based on the dynamic model of the whole cooperative system has been recognized as a critical issue; in fact, this parametrization leads to the definition of task variables for the control and hence to an answer to one of the most frequently asked questions in the field of multi-arm robotics: how to control simultaneously the trajectory of the object, the mechanical stresses (internal forces/moments) acting on the object, load sharing among the arms, and even the external forces/moments on the object. Force decomposition may be a key to solving these problems and has been studied by *Uchiyama* and *Dauchez* [29.11, 12] and *Walker* et al. [29.13] as well as *Bonitz* and *Hsia* [29.14]. In detail, devising

a geometrically clear parametrization of the internal forces/moments acting on the object has been recognized as an important problem; *Williams* and *Khatib* have given a solution to this [29.15, 16]. Several cooperative control schemes based on the sought parameterizations have been designed, including control of motion and force [29.11, 12, 17–19] and impedance/compliance control [29.20–22]. Other approaches include adaptive control [29.23, 24], kinematic control [29.25], task-space regulation [29.26], joint-space control [29.27, 28], and coordinated control [29.29].

Also, the definition of user-oriented task-space variables for coordinated control [29.26] and the development of meaningful performance measures [29.30–33] have been fruitfully investigated in 1990s.

Load sharing among the arms is also an interesting issue on which many papers have been published [29.34–39]. Load sharing may be exploited both for optimal load distribution among arms and for robust holding of the object, when the object is held by the arms without being grasped rigidly. In both cases, anyhow, this becomes a problem of optimization and can be solved by either heuristic [29.40] or mathematical methods [29.41].

Other research efforts have been focused on cooperative handling of multibodied or even flexible objects [29.42–44]. Control of multi-flexible-arm robots has been investigated [29.45, 46], since the merits of flexible-arm robots can be exploited in cooperative systems [29.47], i.e., lightweight structure, intrinsic compliance and hence safety, etc.

Robust holding of an object in the presence of slippage of end-effectors on the object may be achieved as well, if the slippage is detected correctly [29.48].

A more recent control framework for cooperative systems is so-called synchronization control [29.49, 50]; in this class of approaches the control problem is formulated in terms of suitably defined errors accounting for the motion synchronization between the manipulators involved in the cooperative task. As for the nonlinear control of cooperative manipulation systems, efforts have been spent on intelligent control (see, e.g., [29.51] and [29.52], where fuzzy control is exploited to cope with unmodeled dynamics, parametric uncertainties, and disturbances) as well as on the investigation of control strategies in the presence of partial state feedback [29.53].

The problem of the implementation of cooperative control strategies on conventional industrial robots has attracted the increasing interest of the research community. In fact, the control units of industrial robots do

not present all the features needed to implement non-linear torque control schemes, while the integration of force/torque sensing in standard industrial robot control units is often cumbersome and tends to be avoided in industry for many reasons: unreliability, cost, etc. Hence, the rebirth of the early methods, where force sensors were not used (Fujii and Kurono [29.1], Inoue [29.54]), has become attractive for industrial settings. Hybrid position/force control without using force/torque sensors has been successfully implemented [29.55]. Interesting results on the implementation of effective cooperative control strategies for industrial robots are presented, e.g., in [29.56], where a design approach that makes use of tool-based coordinate systems, trajectory generation, and distributed control of multiple robots is presented.

Another interesting aspect, related to the reliability and safety of cooperative manipulation systems, is investigated in [29.57], where the use of nonrigid grippers is considered to avoid large internal forces and, at the same time, to achieve safe manipulation of the object, even in the presence of failures and unpredicted contacts with the external environment.

Finally, it is worth mentioning the strict relationship between problems related to grasping of objects

by fingers/hands (widely described in Chap. 28) and those related to cooperative manipulation. In fact, in both cases, multiple manipulation structures grasp a commonly manipulated object. In multifingered hands only some motion components are transmitted through the contact point to the manipulated object (unilateral constraints), while cooperative manipulation via robotic arms is achieved by rigid (or near-rigid) grasp points and interaction takes place by transmitting all the motion components through the grasping points (bilateral constraints). While many common problems between the two fields can be tackled in a conceptually similar way (e.g., kinetostatic modeling, force control), many other are specific of each of the two application fields (e.g., form and force closure for multifingered hands). Interestingly, a common frame for both cooperative and multifingered manipulation has been proposed [29.25]; namely rolling/sliding of the grasp points on the object is taken into account by modeling the contacts via rotational/prismatic joints; then, the desired manipulator/finger joints trajectories are derived from the desired object motion by using numerical inverse kinematics algorithms.

29.2 Kinematics and Statics

Consider a system composed by M manipulators, each equipped with N_i joints ($i = 1, \dots, M$). Let \mathbf{p}_i denote the (3×1) vector of the position of the i -th end-effector coordinate frame, \mathcal{T}_i , with respect to a common base frame, \mathcal{T} ; let \mathbf{R}_i denote the (3×3) matrix which expresses the orientation of \mathcal{T}_i with respect to the base frame \mathcal{T} .

Both \mathbf{p}_i and \mathbf{R}_i can be expressed as a function of the $(N_i \times 1)$ vector of the joints variables of each manipulator \mathbf{q}_i through the direct kinematics equations

$$\begin{cases} \mathbf{p}_i = \mathbf{p}_i(\mathbf{q}_i), \\ \mathbf{R}_i = \mathbf{R}_i(\mathbf{q}_i). \end{cases} \quad (29.1)$$

Of course, the orientation of the end-effector frame may be expressed in terms of a minimal set of angles, i.e., a set of three Euler angles $\boldsymbol{\phi}_i$. In this case, direct kinematics can be expressed in terms of an operational space vector \mathbf{x}_i as follows

$$\mathbf{x}_i = \mathbf{f}_i(\mathbf{q}_i) = \begin{pmatrix} \mathbf{p}_i(\mathbf{q}_i) \\ \boldsymbol{\phi}_i(\mathbf{q}_i) \end{pmatrix}. \quad (29.2)$$

The linear $\dot{\mathbf{p}}_i$ and angular $\boldsymbol{\omega}_i$ velocity of the i -th end-effector can be collected in the (6×1) generalized velocities vector $\mathbf{v}_i = (\dot{\mathbf{p}}_i^\top \boldsymbol{\omega}_i^\top)^\top$. Then, the differential direct kinematics can be expressed as

$$\mathbf{v}_i = \mathbf{J}_i(\mathbf{q}_i)\dot{\mathbf{q}}_i, \quad (29.3)$$

where the $(6 \times N_i)$ matrix \mathbf{J}_i is the so-called *geometric Jacobian* of the i -th manipulator (see Chap. 1). When the velocity is expressed as the derivative of the operational space vector, differential kinematics takes a formally similar form

$$\dot{\mathbf{x}}_i = \frac{\partial \mathbf{f}_i(\mathbf{q}_i)}{\partial \mathbf{q}_i} \dot{\mathbf{q}}_i = \mathbf{J}_{Ai}(\mathbf{q}_i)\dot{\mathbf{q}}_i, \quad (29.4)$$

where the $(6 \times N_i)$ matrix \mathbf{J}_{Ai} is the so-called *analytical Jacobian* of the i -th manipulator (see Chap. 1).

Let us consider the (6×1) vector of the generalized forces acting at the i -th end-effector

$$\mathbf{h}_i = \begin{pmatrix} \mathbf{f}_i \\ \mathbf{n}_i \end{pmatrix}, \quad (29.5)$$

where f_i and n_i denote the force and moment, respectively. By invoking the principle of virtual work, a relation dual to (29.3) can be derived

$$\tau_i = J^\top(q_i)h_i, \quad (29.6)$$

where τ_i is the $(N_i \times 1)$ vector of the forces/torques acting at the joints of the i -th manipulator.

For the sake of simplicity, consider a system of two cooperative robots (Fig. 29.1) manipulating a common object. Let C be a fixed point of the object (e.g., the center of mass of the object), whose position in the base frame is given by the vector p_C ; moreover, let \mathcal{T}_C be a coordinate frame attached to object with origin in C . The *virtual stick* [29.11, 12] is defined as the vector r_i ($i = 1, 2$) which determines the position of \mathcal{T}_C with respect to \mathcal{T}_i ($i = 1, 2$); it is assumed that r_i behaves as a rigid stick fixed to the i -th end-effector. Hence, each virtual stick, expressed in the frame \mathcal{T}_i (or \mathcal{T}_C), is a constant vector if the commonly grasped object is rigid and tightly grasped by each manipulator. In this case, the direct kinematics of each manipulator can be expressed in terms of a virtual end-effector frame $\mathcal{T}_{S,i} = \mathcal{T}_C$, having the same orientation of \mathcal{T}_C and origin located in $p_{S,i} = p_i + r_i = p_C$. Hence, the position and orientation at the tip of each virtual stick are given by ($i = 1, 2$)

$$p_{S,i} = p_C, \quad R_{S,i} = R_C.$$

The set of Euler angles corresponding to $R_{S,i}$ will be denoted by the (3×1) vector $\phi_{S,i}$. Hereafter, it is assumed that the object grasped by the two manipulators can be considered rigid (or nearly rigid) and tightly (or nearly tightly) attached to each end-effector. Hence, the displacements between the above defined frames can be

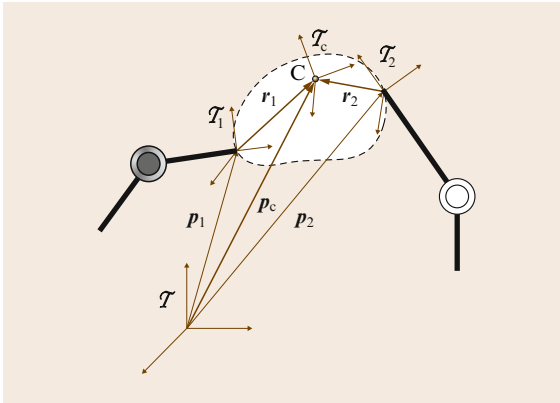


Fig. 29.1 Grasp geometry for a two-manipulator cooperative system manipulating a common object

considered null or negligible. Otherwise, if the manipulated object is subject to deformation (e.g., in the case of a flexible object) and/or the grasp is not tight (e.g., in the case of compliant grippers) the above defined coordinate frames, will be subject to nonnegligible displacements with respect to each other.

Let $h_{S,i}$ denote the vector of generalized forces acting at the tip of the i -th virtual stick; it can be easily verified that the following equation holds

$$h_{S,i} = \begin{pmatrix} I_3 & O_3 \\ -S(r_i) & I_3 \end{pmatrix} h_i = W_i h_i, \quad (29.7)$$

where O_l and I_l denote, respectively, the null matrix and the identity matrix of $(l \times l)$ dimensions, and $S(r_i)$ is the (3×3) skew-symmetric matrix operator performing the cross product. It is worth noticing that W_i is always full rank.

By invoking the virtual works principle, a relation dual to (29.7) can easily be derived

$$v_i = \begin{pmatrix} I_3 & S(r_i) \\ O_3 & I_3 \end{pmatrix} v_{S,i} = W_i^\top v_{S,i}, \quad (29.8)$$

where $v_{S,i}$ is the generalized velocity vector of the virtual stick endpoint. When $r_i = 0$, it is $W_i = I_6$; in other words, if the end-effector kinematics of each manipulator is referred to the corresponding virtual stick (or the object reduces to point), the forces and velocities at the two end-effectors coincide with their counterparts referred at the virtual sticks.

29.2.1 Symmetric Formulation

The kinetostatic formulation proposed by Uchiyama and Dauchez [29.12], i.e., the so-called *symmetric formulation*, is based on kinematic and static relationships between generalized forces/velocities acting at the object and their counterparts acting at the manipulators end-effectors (or at the virtual sticks endpoints).

Let us define first the *external forces* as the (6×1) vector of generalized forces given by

$$h_E = h_{S,1} + h_{S,2} = W_S h_S, \quad (29.9)$$

with $W_S = (I_6 \ I_6)$ and $h_S = (h_{S,1}^\top \ h_{S,2}^\top)^\top$; in other words, h_E represents the vector of generalized forces causing the object's motion. From (29.7) and (29.9) it follows that h_E can be expressed in terms of end-effector forces as well

$$h_E = W_1 h_1 + W_2 h_2 = W h, \quad (29.10)$$

with $W = (W_1 \ W_2)$ and $h = (h_1^\top \ h_2^\top)^\top$. It can be recognized that W_S (W) is a (6×12) matrix, having

a six-dimensional range space and a six-dimensional null space, describing the geometry of the grasp, and thus is usually termed the *grasp matrix*.

The inverse solution to (29.9) yields

$$\mathbf{h}_S = \mathbf{W}_S^\dagger \mathbf{h}_E + \mathbf{V}_S \mathbf{h}_I = \mathbf{U}_S \mathbf{h}_O, \quad (29.11)$$

where \mathbf{W}_S^\dagger denotes the Moore–Penrose pseudoinverse of \mathbf{W}_S

$$\mathbf{W}_S^\dagger = \frac{1}{2} \begin{pmatrix} \mathbf{I}_6 \\ \mathbf{I}_6 \end{pmatrix}. \quad (29.12)$$

\mathbf{V}_S is a matrix whose columns are a basis of the null space of \mathbf{W}_S , e.g.,

$$\mathbf{V}_S = \begin{pmatrix} -\mathbf{I}_6 \\ \mathbf{I}_6 \end{pmatrix}, \quad (29.13)$$

$$\mathbf{h}_O = (\mathbf{h}_E^\top \mathbf{h}_I^\top)^\top, \text{ and}$$

$$\mathbf{U}_S = (\mathbf{W}_S^\dagger \mathbf{V}_S). \quad (29.14)$$

The second term on the right-hand side of (29.11), i.e., $\mathbf{V}_S \mathbf{h}_I$, represents a vector of generalized forces, referred at the tip of the virtual sticks, which lies in the null space of \mathbf{W}_S ; thus, such forces do not contribute to external forces. Hence, the (6×1) vector \mathbf{h}_I is a generalized force which does not contribute to the object's motion. Therefore, it represents internal loading of the object (i.e., mechanical stresses) and is termed the *internal forces* vector [29.12]. A similar argument can be used for equation (29.10), whose inverse solution is

$$\mathbf{h} = \mathbf{W}^\dagger \mathbf{h}_E + \mathbf{V} \mathbf{h}_I = \mathbf{U} \mathbf{h}_O, \quad (29.15)$$

where

$$\mathbf{U} = (\mathbf{W}^\dagger \mathbf{V}). \quad (29.16)$$

In [29.13] it has been shown that the first term on the right-hand side of (29.15) represents only contributions to external forces if the pseudoinverse of the grasp matrix is properly defined, i.e.,

$$\mathbf{W}^\dagger = \begin{pmatrix} \frac{1}{2} \mathbf{I}_3 & \mathbf{O}_3 \\ \frac{1}{2} \mathbf{S}(\mathbf{r}_1) & \frac{1}{2} \mathbf{I}_3 \\ \frac{1}{2} \mathbf{I}_3 & \mathbf{O}_3 \\ \frac{1}{2} \mathbf{S}(\mathbf{r}_2) & \frac{1}{2} \mathbf{I}_3 \end{pmatrix}. \quad (29.17)$$

As \mathbf{V}_S in (29.11), the columns of \mathbf{V} span the null space of \mathbf{W} , and can be chosen as [29.30]

$$\mathbf{V} = \begin{pmatrix} -\mathbf{I}_3 & \mathbf{O}_3 \\ -\mathbf{S}(\mathbf{r}_1) & -\mathbf{I}_3 \\ \mathbf{I}_3 & \mathbf{O}_3 \\ \mathbf{S}(\mathbf{r}_2) & \mathbf{I}_3 \end{pmatrix}. \quad (29.18)$$

A different parametrization of the inverse solutions to (29.9) and (29.10) can be expressed, respectively, as

$$\mathbf{h}_S = \mathbf{W}_S^\dagger \mathbf{h}_E + (\mathbf{I}_{12} - \mathbf{W}_S^\dagger \mathbf{W}_S) \mathbf{h}_S^* \quad (29.19)$$

and

$$\mathbf{h} = \mathbf{W}^\dagger \mathbf{h}_E + (\mathbf{I}_{12} - \mathbf{W}^\dagger \mathbf{W}) \mathbf{h}^*, \quad (29.20)$$

where \mathbf{h}_S^* (\mathbf{h}^*) is an arbitrary (12×1) vector of generalized forces acting at the tip of the i -th virtual stick (i -th end-effector) projected onto the null space of \mathbf{W}_S (\mathbf{W}) via $\mathbf{I}_{12} - \mathbf{W}_S^\dagger \mathbf{W}_S$ ($\mathbf{I}_{12} - \mathbf{W}^\dagger \mathbf{W}$).

By exploiting the principle of virtual work, the mappings between generalized velocities, dual to those derived above, can be established. In detail, the mapping dual to (29.11) is:

$$\mathbf{v}_O = \mathbf{U}_S^\top \mathbf{v}_S, \quad (29.21)$$

where $\mathbf{v}_S = (\mathbf{v}_{S,1}^\top \mathbf{v}_{S,2}^\top)^\top$, $\mathbf{v}_O = (\mathbf{v}_E^\top \mathbf{v}_I^\top)^\top$. The vector \mathbf{v}_E can be interpreted as the absolute velocity of the object, while \mathbf{v}_I represents the relative velocity between the two coordinate frames $\mathcal{T}_{S,1}$ and $\mathcal{T}_{S,2}$ attached to the tips of the virtual sticks [29.12]; this vector is null when the manipulated object is rigid and rigidly grasped. In a similar way, from (29.15), the following mapping can be devised

$$\mathbf{v}_O = \mathbf{U}^\top \mathbf{v}, \quad (29.22)$$

where $\mathbf{v} = (\mathbf{v}_1^\top \mathbf{v}_2^\top)^\top$.

Moreover, a set of position and orientation variables corresponding to \mathbf{v}_E and \mathbf{v}_I can be defined as follows [29.12, 25]

$$\mathbf{p}_E = \frac{1}{2}(\mathbf{p}_{S,1} + \mathbf{p}_{S,2}), \quad \mathbf{p}_I = \mathbf{p}_{S,2} - \mathbf{p}_{S,1}, \quad (29.23)$$

$$\mathbf{R}_E = \mathbf{R}_1 \mathbf{R}^1(k_{21}^1, \vartheta_{21}/2), \quad \mathbf{R}_I^1 = \mathbf{R}_2^1, \quad (29.24)$$

where $\mathbf{R}_2^1 = \mathbf{r}_1^\top \mathbf{R}_2$ is the matrix expressing the orientation of \mathcal{T}_2 with respect to the axes of \mathcal{T}_1 , while k_{21}^1 and ϑ_{21} are, respectively, the equivalent unit vector (expressed with respect to \mathcal{T}_1) and rotation angle that determine the mutual orientation expressed by \mathbf{R}_2^1 . Hence, \mathbf{R}_E expresses a rotation about k_{21}^1 by an angle which is half the angle needed to align \mathcal{T}_2 with \mathcal{T}_1 .

In turn, if the orientation variables are expressed in terms of Euler angles, a set of operational space variables can be defined as

$$\mathbf{x}_E = \begin{pmatrix} p_E \\ \phi_E \end{pmatrix}, \quad \mathbf{x}_I = \begin{pmatrix} p_I \\ \phi_I \end{pmatrix}, \quad (29.25)$$

where

$$\phi_E = \frac{1}{2}(\phi_{S,1} + \phi_{S,2}), \quad \phi_I = \phi_{S,2} - \phi_{S,1}. \quad (29.26)$$

It must be remarked, however, that the definitions in (29.26) keep a clear geometric meaning only if the orientation displacements between the virtual stick frames are small. In this case, as shown in [29.11, 12], the corresponding operational space velocities, $\dot{\mathbf{x}}_E$ and $\dot{\mathbf{x}}_I$, correspond to \mathbf{v}_E and \mathbf{v}_I , respectively, with good approximation. Otherwise, if the orientation displacements become large, the variables defined in (29.26) do not represent any meaningful quantities, and other orientation representations have to be adopted, e.g., the unit quaternion (see Chap. 1 for a general introduction to quaternions and Sect. 29.3 for an application to cooperative robots kinematics).

Finally, by using (29.15) and (29.22), together with equations (29.3) and (29.6), the kinetostatic mappings between the force/velocities at the object and their counterparts in the joint space of the manipulators can be given

$$\boldsymbol{\tau} = \mathbf{J}_O^\top \mathbf{h}_O, \quad (29.27)$$

$$\mathbf{v}_O = \mathbf{J}_O \dot{\mathbf{q}}, \quad (29.28)$$

where $\boldsymbol{\tau} = (\tau_1^\top \tau_2^\top)^\top$, $\mathbf{q} = (q_1^\top q_2^\top)^\top$, and

$$\mathbf{J}_O = \mathbf{U}^\top \mathbf{J}, \quad \mathbf{J} = \begin{pmatrix} \mathbf{J}_1 & \mathbf{O}_6 \\ \mathbf{O}_6 & \mathbf{J}_2 \end{pmatrix}. \quad (29.29)$$

Formally similar mappings can be established in terms of operational space velocities $\dot{\mathbf{x}}_E$ and $\dot{\mathbf{x}}_I$ [29.11, 12] and the corresponding operational space forces/moments.

29.2.2 Multifingered Manipulation

Hereafter some connections between the field of multi-arm cooperative manipulation, described in this chapter, and multifingered manipulation, described in Chap. 28, are briefly outlined, focusing on the kinetostatics of the two classes of manipulation systems.

Both in the case of multi-arm cooperative systems and of multifingered manipulation, two or more manipulation structures grasp a commonly manipulated object.

Cooperative manipulation via multiple robotic arms is achieved by rigidly grasping the object (e.g., via rigid

fixtures), and thus interaction takes place by exchanging both forces and moments at the grasp points. In other words, all the translational and rotational motion components are transmitted through the grasp points.

On the other side, when a multifingered hand manipulates an object, only some components of the motion are transmitted through the contact points. This is effectively modeled via properly defined constraint matrices, whose expression depends on the type of contact. In other words, constraint matrices act as filters selecting the components of the motion transmitted through each contact. In fact, as shown in Chap. 28, it is convenient to think of the object–finger contact point as twofold: a point on the tip of the finger and a point on the object. Hence, two generalized velocity vectors are defined for the i -th contact point (both expressed with respect to frame \mathcal{T}_i): the velocity of the contact point on the hand, $\mathbf{v}_{h,i}^i$, and the velocity of the contact point on the object $\mathbf{v}_{o,i}^i$. The corresponding dual generalized force vectors are $\mathbf{h}_{h,i}^i$ and $\mathbf{h}_{o,i}^i$, respectively. A contact model is then defined via the $(m_i \times 6)$ constraint matrix \mathbf{H}_i , selecting m_i velocity components transmitted through the contact ($\mathbf{v}_{t,i}^i$), i. e.,

$$\mathbf{v}_{t,i}^i = \mathbf{H}_i \mathbf{v}_{h,i}^i = \mathbf{H}_i \mathbf{v}_{o,i}^i. \quad (29.30)$$

The relation dual to (29.30) is

$$\mathbf{H}_i^\top \mathbf{h}_{t,i}^i = \mathbf{h}_{h,i}^i = \mathbf{h}_{o,i}^i, \quad (29.31)$$

where $\mathbf{h}_{t,i}^i$ is the vector of the transmitted generalized forces. Hence, (29.10) can be rewritten as

$$\mathbf{h}_E = \mathbf{W}_1 \bar{\mathbf{R}}_1 \mathbf{H}_1^\top \mathbf{h}_{t,1}^1 + \mathbf{W}_2 \bar{\mathbf{R}}_2 \mathbf{H}_2^\top \mathbf{h}_{t,2}^2, \quad (29.32)$$

where $\bar{\mathbf{R}}_i = \text{diag}\{\mathbf{R}_i, \mathbf{R}_i\}$. Hence, a conceptually similar kinetostatic analysis can be developed, leading to the concept of external and internal forces (and related kinematic quantities) as well.

However, it must be remarked that, while in the case of cooperative manipulators, tightly grasping the object, internal stresses are usually undesirable effects (unless controlled squeezing of a deformable object has to be achieved), in multifingered hands suitably controlled internal forces are useful to guarantee a firm grasp even in the presence of external loading applied to the object (see, e.g., the problem of form and force closure discussed in Chap. 28).

29.3 Cooperative Task Space

The task-oriented formulation for the definition of the *cooperative task space* originally proposed in [29.25,26] is here briefly reviewed. By starting from (29.23) and (29.24), this formulation defines directly the task variables in terms of the *absolute* and *relative* motion of the cooperative system, which can be directly computed from the position/orientation of the end-effector coordinate frames.

Let us define as *absolute coordinate frame*, \mathcal{T}_a , the frame whose position in the base frame is given by the position vector \mathbf{p}_a (*absolute position*)

$$\mathbf{p}_a = \frac{1}{2}(\mathbf{p}_1 + \mathbf{p}_2). \quad (29.33)$$

The orientation of \mathcal{T}_a with respect to the base frame (*absolute orientation*) is defined by means of the rotation matrix \mathbf{R}_a

$$\mathbf{R}_a = \mathbf{R}_1 \mathbf{R}^1(k_{21}^1, \vartheta_{21}/2). \quad (29.34)$$

Of course, the sole absolute variables cannot uniquely specify the cooperative motion, since 12 variables are required, e.g., for a two-arm system. Hence, in order to devise a complete description of the system's configuration, the position/orientation of each manipulator relative to the other manipulators in the system must be considered. In detail, for a two-arm system, the *relative position* between the manipulators is defined as

$$\mathbf{p}_r = \mathbf{p}_2 - \mathbf{p}_1, \quad (29.35)$$

while the *relative orientation* between the two end-effector frames is expressed via the following rotation matrix

$$\mathbf{R}_r^1 = \mathbf{R}_1^\top \mathbf{R}_2 = \mathbf{R}_2^1. \quad (29.36)$$

The variables \mathbf{p}_a , \mathbf{R}_a , \mathbf{p}_r , and \mathbf{R}_r^1 define the *cooperative task space*. Remarkably, \mathbf{R}_a and \mathbf{R}_r^1 coincide with \mathbf{R}_E and \mathbf{R}_I^1 , respectively.

It is worth pointing out a useful feature of the above defined cooperative task space formulation. In fact, it can be recognized that definitions (29.33–29.36) are not based on any special assumption on the nature of the manipulated object and/or the grasp. In other words, the cooperative task space variables can be effectively used to describe cooperative systems manipulating non-rigid objects and/or characterized by a nonrigid grasp. Also, the same task-space variables can be used to describe a pure motion coordination task, i.e., when the arms are required to perform a coordinated motion without physically interacting via a commonly manipulated

object. When the manipulators hold a rigid object (or a deformable object for which deformations are not commanded), then relative position and orientation are to be kept constant. Otherwise, if a relative motion between the end-effectors is allowed (or commanded), then \mathbf{p}_r^1 and \mathbf{R}_r^1 may vary according to the effective (commanded) relative motion.

As shown in [29.25,26], absolute

$$\dot{\mathbf{p}}_a = \frac{1}{2}(\dot{\mathbf{p}}_1 + \dot{\mathbf{p}}_2), \quad \boldsymbol{\omega}_a = \frac{1}{2}(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2), \quad (29.37)$$

and relative

$$\dot{\mathbf{p}}_r = (\dot{\mathbf{p}}_2 - \dot{\mathbf{p}}_1), \quad \boldsymbol{\omega}_r = \boldsymbol{\omega}_2 - \boldsymbol{\omega}_1, \quad (29.38)$$

linear and angular velocities can be readily derived, as well as their dual variables (i.e., absolute/relative forces and moments)

$$\mathbf{f}_a = \mathbf{f}_1 + \mathbf{f}_2, \quad \mathbf{n}_a = \mathbf{n}_1 + \mathbf{n}_2, \quad (29.39)$$

$$\mathbf{f}_r = \frac{1}{2}(\mathbf{f}_2 - \mathbf{f}_1), \quad \mathbf{n}_r = \frac{1}{2}(\mathbf{n}_2 - \mathbf{n}_1). \quad (29.40)$$

Kinetostatic mappings, analogous to those derived in the previous section, can be established between linear/angular velocities (force/moments) and their counterparts defined at the end-effectors (or joints) level of each manipulator [29.25,26].

Remarkably, the variables defined by the symmetric formulation and those used in the task-oriented formulation are related via simple mappings. In fact, forces (angular velocities, orientation variables) always coincide in the two formulations, while moments (linear velocities, position variables) coincide only when the object reduces to a point, or when the kinematics of each manipulator is referred to the tip of the corresponding virtual stick.

As shown in the following example, in the case of planar cooperative systems, the definition of the cooperative task-space variables is straightforward.

Example 29.1 (cooperative task-space variables for a planar two-arm system): For a planar two-arm system the i -th end-effector coordinates can be defined via the (3×1) vector

$$\mathbf{x}_i = \begin{pmatrix} \mathbf{p}_i \\ \varphi_i \end{pmatrix}, \quad i = 1, 2,$$

where \mathbf{p}_i is the (2×1) vector of the i -th end-effector position in the plane and φ_i is the angle describing its

orientation (i.e., the rotation of the end-effector frame about an axis orthogonal to the plane). Hence, the task-space variables can be readily defined as

$$\mathbf{x}_a = \frac{1}{2}(\mathbf{x}_1 + \mathbf{x}_2), \quad (29.41)$$

$$\mathbf{x}_r = \mathbf{x}_2 - \mathbf{x}_1, \quad (29.42)$$

since the orientation of each end-effector is simply represented by an angle.

In the spatial case, the definition of the orientation variables in terms of Euler angles, as in (29.34) and (29.36), is somewhat critical; in fact, since \mathcal{T}_1 and \mathcal{T}_2 do not coincide in general, orientation displacements are likely to be large and definitions analogous to (29.26) are not correct. Hence, geometrically meaningful orientation representations have to be adopted, e.g., the unit quaternion (see Chap. 1). In detail, by following

the approach in [29.26], the orientation variables can be defined as follows. Let

$$\mathcal{Q}_k^1 = \{\eta_k, \epsilon_k^1\} = \left\{ \cos \frac{\vartheta_{21}}{4}, k_{21}^1 \sin \frac{\vartheta_{21}}{4} \right\} \quad (29.43)$$

denote the unit quaternion extracted from $\mathbf{R}^1(k_{21}^1, \vartheta_{21}/2)$; let, also, $\mathcal{Q}_1 = \{\eta_1, \epsilon_1\}$ and $\mathcal{Q}_2 = \{\eta_2, \epsilon_2\}$ denote the unit quaternions extracted from \mathbf{R}_1 and \mathbf{R}_2 , respectively. Then, the absolute orientation can be expressed in terms of quaternion product as follows

$$\mathcal{Q}_a = \{\eta_a, \epsilon_a\} = \mathcal{Q}_1 * \mathcal{Q}_k^1, \quad (29.44)$$

while the relative orientation can be expressed as the quaternion product

$$\mathcal{Q}_r^1 = \{\eta_r, \epsilon_r^1\} = \mathcal{Q}_1^{-1} * \mathcal{Q}_2, \quad (29.45)$$

where $\mathcal{Q}_1^{-1} = \{\eta_1, -\epsilon_1\}$ (i.e., the conjugate of \mathcal{Q}_1) represents the unit quaternion extracted from \mathbf{R}_1^\top .

29.4 Dynamics and Load Distribution

The equations of motion of the i -th manipulator in a cooperative manipulation system are given by

$$\mathbf{M}_i(\mathbf{q}_i) \ddot{\mathbf{q}}_i + \mathbf{c}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i) = \boldsymbol{\tau}_i - \mathbf{J}_i^\top(\mathbf{q}_i) \mathbf{h}_i, \quad (29.46)$$

where $\mathbf{M}_i(\mathbf{q}_i)$ is the symmetric positive-definite inertia matrix and $\mathbf{c}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)$ is the vector of the forces/torques due to the centrifugal, Coriolis, gravity, and friction effects. The model can be expressed in compact form as

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau} - \mathbf{J}^\top(\mathbf{q}) \mathbf{h}, \quad (29.47)$$

where the matrices are block-diagonal (e.g., $\mathbf{M} = \text{blockdiag}\{\mathbf{M}_1, \mathbf{M}_2\}$) and the vectors are stacked (e.g., $\mathbf{q} = (\mathbf{q}_1^\top \mathbf{q}_2^\top)^\top$).

The object's motion is described by the classical Newton–Euler equations of rigid body

$$\mathbf{M}_E(\mathbf{R}_E) \dot{\mathbf{v}}_E + \mathbf{c}_E(\mathbf{R}_E, \boldsymbol{\omega}_E) \mathbf{v}_E = \mathbf{h}_E = \mathbf{W} \mathbf{h}, \quad (29.48)$$

where \mathbf{M}_E is the inertia matrix of the object and \mathbf{c}_E collects the nonlinear components of the inertial forces/moments (i.e., gravity, centrifugal and Coriolis forces/moments).

The above equations must be completed by imposing the closed-chain constraints arising from the kinematic coupling between the two manipulators through the commonly manipulated rigid object. The constraints can

be expressed by imposing a null internal velocity vector in the mapping (29.21)

$$\mathbf{v}_1 = \mathbf{V}_S^\top \mathbf{v}_S = \mathbf{v}_{S,1} - \mathbf{v}_{S,2} = \mathbf{0}, \quad (29.49)$$

which can be expressed, by using (29.8) and (29.22), in terms of end-effector velocities (where the notation $\mathbf{W}_i^{-\top}$ stands for $(\mathbf{W}_i^\top)^{-1}$)

$$\mathbf{V}^\top \mathbf{v} = \mathbf{W}_1^{-\top} \mathbf{v}_1 - \mathbf{W}_2^{-\top} \mathbf{v}_2 = \mathbf{0}, \quad (29.50)$$

and, finally, in terms of joint velocities

$$\mathbf{V}^\top \mathbf{J}(\mathbf{q}) \dot{\mathbf{q}} = \mathbf{W}_1^{-\top} \mathbf{J}_1(\mathbf{q}_1) \dot{\mathbf{q}}_1 - \mathbf{W}_2^{-\top} \mathbf{J}_2(\mathbf{q}_2) \dot{\mathbf{q}}_2 = \mathbf{0}. \quad (29.51)$$

Equations (29.47), (29.48), and (29.51) represent a constrained dynamical model of the cooperative system in the joint space; $N_1 + N_2$ generalized coordinates (i.e., \mathbf{q}_1 and \mathbf{q}_2) are related to each other by the six algebraic closed-chain constraints (29.51). This implies that the total number of degrees of freedom is $N_1 + N_2 - 6$ and the model has the form of a set of differential-algebraic equations.

29.4.1 Reduced-Order Models

The above derived dynamic model incorporates a set of closed-chain constraints, which reduces the number of independent generalized coordinates to $N_1 + N_2 - 6$.

Hence, it is expected that, by eliminating six equations via the constraints (29.51), a reduced-order model can be obtained. Early work on reduced-order modeling of closed chains can be, e.g., found in [29.58]. Later, in [29.59, 60], this problem has been tackled by first deriving, from (29.47), (29.48), and (29.51), a joint space model of the whole closed chain in the form

$$\mathbf{M}_C(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}_C(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{D}_C(\mathbf{q})\boldsymbol{\tau}, \quad (29.52)$$

where \mathbf{M}_C , \mathbf{D}_C , and \mathbf{c}_C depend on the dynamics of the manipulators and the object, as well as on the geometry of the grasp. The above model can be integrated, once the joint torques vector $\boldsymbol{\tau}$ is specified over an assigned time interval, to solve for the joint variables \mathbf{q} (*forward dynamics*). However, the model cannot be used to find $\boldsymbol{\tau}$ from assigned \mathbf{q} , $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ (*inverse dynamics*), since the $((N_1 + N_2) \times (N_1 + N_2))$ matrix \mathbf{D}_C is not full rank [29.60], and thus the inverse dynamics problem turns out to be underspecified.

In order to find a reduced-order model, composed by $N_1 + N_2 - 6$ equations, a $((N_1 + N_2 - 6) \times 1)$ pseudoveLOCITY vector has to be considered:

$$\mathbf{v} = \mathbf{B}(\mathbf{q})\dot{\mathbf{q}}, \quad (29.53)$$

where the $((N_1 + N_2 - 6) \times (N_1 + N_2))$ matrix $\mathbf{B}(\mathbf{q})$ is selected so that $(\mathbf{A}^\top(\mathbf{q}) \mathbf{B}^\top(\mathbf{q}))^\top$ is nonsingular and

$$\mathbf{A}(\mathbf{q}) = \mathbf{W}_2^\top \mathbf{V}^\top(\mathbf{q}).$$

Then, the reduced-order model can be written in terms of the variables \mathbf{q} , \mathbf{v} , and $\dot{\mathbf{v}}$ as

$$\boldsymbol{\Sigma}^\top(\mathbf{q})\mathbf{M}_C(\mathbf{q})\boldsymbol{\Sigma}(\mathbf{q})\dot{\mathbf{v}} + \boldsymbol{\Sigma}^\top(\mathbf{q})\mathbf{c}_R(\mathbf{q}, \mathbf{v}) = \boldsymbol{\Sigma}^\top(\mathbf{q})\boldsymbol{\tau}, \quad (29.54)$$

where $\boldsymbol{\Sigma}$ is an $((N_1 + N_2) \times (N_1 + N_2 - 6))$ matrix such that

$$\begin{pmatrix} \mathbf{A} \\ \mathbf{B} \end{pmatrix}^{-1} = (\boldsymbol{\Pi} \boldsymbol{\Sigma}),$$

and \mathbf{c}_R depends on \mathbf{c}_C , $\boldsymbol{\Sigma}$, and $\boldsymbol{\Pi}$ [29.60]. The reduced-order model can be used for computing the forward dynamics; in this case, however, the problem of expressing a reduced set of *pseudocoordinates* related to \mathbf{v} in the numerical integration has to be considered. Since $\boldsymbol{\Sigma}^\top$ is nonsquare, the inverse dynamics problem still admits infinitely many solutions in terms of $\boldsymbol{\tau}$. However, this does not prevent the application model (29.54) to cooperative manipulators control (see, e.g., the decoupled control architecture proposed in [29.59, 60]).

29.4.2 Load Distribution

The problem of load sharing in multi-arm robotic systems is that of distributing the load among the arms composing the system (e.g., a strong arm may share the load more than a weak one). This is possible because a multi-arm system has redundant actuators; in fact, if a robotic arm is equipped with the number of actuators strictly needed for supporting the load, no optimization of load distribution is possible. In this section, this problem is presented according to the results in [29.35–41].

We can introduce a load-sharing matrix in the framework adopted for presenting the kinematics of the cooperative manipulators. By replacing the Moore–Penrose inverse in equation (29.11) with a suitably defined generalized inverse, \mathbf{W}_S^- , the following expression can be devised for the generalized forces at the tip of the virtual sticks

$$\mathbf{h}_S = \mathbf{W}_S^- \mathbf{h}_E + \mathbf{V}_S \mathbf{h}'_I, \quad (29.55)$$

where

$$\mathbf{W}_S^- = \begin{pmatrix} \mathbf{L} \\ \mathbf{I}_6 - \mathbf{L} \end{pmatrix}^\top, \quad (29.56)$$

and the matrix \mathbf{L} is the so-called load sharing matrix. It can be easily proved that the nondiagonal elements of \mathbf{L} only yield a \mathbf{h}_S vector in the null space of \mathbf{W}_S , that is, the space of internal forces/moments. Therefore, without losing generality, let us choose \mathbf{L} such that

$$\mathbf{L} = \text{diag}\{\boldsymbol{\lambda}\}, \quad (29.57)$$

where the vector $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_6)^\top$ collects a set of constants λ_i representing the load-sharing coefficients.

The problem is that of properly tuning the load-sharing coefficients to ensure correct cooperative manipulation of the object by the arms. In order to answer this question, it must be noticed that, by combining equations (29.11) and (29.55), the following relation can be obtained

$$\mathbf{h}_I = \mathbf{V}_S^\dagger (\mathbf{W}_S^- - \mathbf{W}_S^\dagger) \mathbf{h}_E + \mathbf{h}'_I, \quad (29.58)$$

which, bearing in mind that only \mathbf{h}_E and \mathbf{h}_S are really existing forces/moments, indicates that

- \mathbf{h}_I , \mathbf{h}'_I , and λ_i can be thought of as *artificial* parameters, introduced for better understanding of the manipulation process;
- \mathbf{h}'_I and λ_i are not independent; the concept of internal forces/moments and the concept of load sharing are mathematically mixed with each other.

Therefore, tuning the load-sharing coefficients or choosing suitable internal forces/moments is strictly equivalent from the mathematical (and, also, from the performance) point of view. Only one variable among \mathbf{h}_I , \mathbf{h}'_I , and λ is independent; hence, any of those redundant parameters, to be optimized for load sharing, can be exploited. This is more generally stated in [29.38, 39]. In [29.41], internal forces/moments \mathbf{h}_I are tuned, both for the sake of simplicity and for consistency with the adopted control laws.

A very relevant problem related to load sharing is that of robust holding, i.e., the problem of determining forces/moments applied to object by the arms, \mathbf{h}_S , required to keep the grasp even in the presence of disturbing external forces/moments. This problem can be solved by tuning the internal forces/moments (or the load-sharing coefficients, of course) and is addressed, e.g., in [29.40], where conditions to keep the grasp are expressed via end-effector forces/moments. Namely, by replacing \mathbf{h}_S in (29.55) into the equations expressing the grasp conditions, a set of linear inequalities for both \mathbf{h}'_I and λ are obtained:

$$\mathbf{A}_L \mathbf{h}'_I + \mathbf{B}_L \lambda < \mathbf{c}_L, \quad (29.59)$$

where \mathbf{A}_L and \mathbf{B}_L are 6×6 matrices and \mathbf{c}_L is a (6×1) vector. In [29.40], a solution λ for the inequality is obtained heuristically. The above inequality can be transformed into another inequality expressed with respect to

\mathbf{h}_I , of course; however, λ is fitter to such heuristic algorithm because it can be understood intuitively. The same problem may be solved mathematically, by introducing an objective function to be optimized; in this way, the problem is recast in the framework of mathematical programming. To this purpose, let us choose a quadratic cost function of \mathbf{h}_I to be minimized

$$f = \mathbf{h}_I^\top \mathbf{Q} \mathbf{h}_I, \quad (29.60)$$

where \mathbf{Q} is a (6×6) positive-definite matrix. The above cost function can be seen as a (pseudo)energy to be dissipated by the joint actuators: i.e., the arms dissipate electrical energy in the actuators to yield the internal forces/moments \mathbf{h}_I . A solution to the quadratic programming problem (29.60) can be found, e.g., in [29.41].

For further insights on the problem of robust holding in the framework of multifingered manipulation, the reader is referred to Chap. 28.

Interestingly, in [29.36] the problem of load distribution is formulated in such a way to take into account manipulators dynamics at the joint level. This approach allows the expression of the load-sharing problem directly in terms of joints actuators torques; at this level, different subtask performance indexes can be used, in a similar way as was done to solve the inverse kinematics of redundant manipulators, to achieve load distribution solutions.

29.5 Task-Space Analysis

As for single-arm robotic systems, a major issue in cooperative manipulation is that of task-space performance evaluation via the definition of suitable manipulability ellipsoids. These concepts have been extended to multi-arm robotic systems in [29.30]. Namely, by exploiting the kinetostatic formulation in Sect. 29.3, velocity and force manipulability ellipsoids are defined by regarding the whole cooperative system as a mechanical transformer from the joint space to the cooperative task space. Since the construction of force/velocity ellipsoids involves nonhomogeneous quantities (i.e., forces and moments, linear and angular velocities), special attention must be paid to the definition of such concepts [29.14, 61, 62]. Also, as for the ellipsoids involving internal forces, a major issue is that of physically meaningful parametrization of the internal forces (see, e.g., the work in [29.15, 16]).

In detail, by following the approach in [29.30], the *external force manipulability ellipsoid* can be defined by

the following scalar equation

$$\mathbf{h}_E^\top (\mathbf{J}_E \mathbf{J}_E^\top) \mathbf{h}_E = 1, \quad (29.61)$$

where $\mathbf{J}_E = \mathbf{W}^{\dagger\top} \mathbf{J}$. The *external velocity manipulability ellipsoid* is defined dually by the following scalar equation

$$\mathbf{v}_E^\top (\mathbf{J}_E \mathbf{J}_E^\top)^{-1} \mathbf{v}_E = 1. \quad (29.62)$$

In the case of dual-arm systems the *internal force manipulability ellipsoid* can be defined as

$$\mathbf{h}_I^\top (\mathbf{J}_I \mathbf{J}_I^\top) \mathbf{h}_I = 1, \quad (29.63)$$

where $\mathbf{J}_I = \mathbf{V}^\top \mathbf{J}$. Also, the *internal velocity ellipsoid* can be defined, via kinetostatic duality, as

$$\mathbf{v}_I^\top (\mathbf{J}_I \mathbf{J}_I^\top)^{-1} \mathbf{v}_I = 1. \quad (29.64)$$

The internal force/velocity ellipsoids can be defined in the case of cooperative systems composed of more than

two manipulators by considering one pair of interacting end-effectors at a time [29.30].

The manipulability ellipsoids can be seen as performance measures aimed at determining the arms' attitude to cooperate in a given system's configuration. Also, as for the single-arm systems, the manipulability ellipsoids can be used to determine optimal postures for redundant multi-arm systems.

Besides the above described approach, two other main approaches have been proposed to analyze

the manipulability of cooperative multi-arm systems: the task-oriented manipulability measure [29.31] and polytopes [29.32]. Moreover, in [29.33] a systematic approach to perform dynamic analysis of multi-arm systems is presented. Namely, the concept of dynamic manipulability ellipsoid is extended to multi-arm systems, and indexes of the system's capability of generating object accelerations along assigned task-space directions are devised.

29.6 Control

When a cooperative multi-arm system is employed for the manipulation of a common object, it is important to control both the absolute motion of the held object and the internal stresses applied to it. Hence, most of the control approaches to cooperative robotic systems can be classified as force/motion control schemes, in that they decompose the control action in a motion control loop, aimed at tracking of the desired object motion, and a force control loop, aimed at controlling the internal loading of the object.

Early approaches to the control of cooperative systems are based on the *master/slave* concept [29.2]. Namely, the cooperative system is decomposed into:

- a master arm, which is in charge of imposing the absolute motion of the object; hence, the master arm is position controlled so as to achieve accurate and robust tracking of position/orientation reference trajectories, in the face of external disturbances (e.g., forces due to the interaction with the other cooperating arms); in other words, the master arm is controlled so as to have a *stiff* behavior;
- the slave arms, which are force controlled so as to achieve a *compliant* behavior with respect to the interaction forces; hence, it is expected that the slave arms are capable of following (as smoothly as possible) the motion imposed by the master arm.

A natural evolution of the above approach is the so-called *leader-follower* [29.58], where the follower arms reference motion is computed via closed-chain constraints.

However, such approaches suffered from implementation issues, mainly due to the fact that the compliance of the slave arms has to be very large, so as to follow the motion imposed by the master arm smoothly. Also, a difficulty arises when the roles of master and slave have to be assigned to the arms for a given coopera-

tive task, since the master/slave modes may need to be dynamically changed during the task execution.

Hence, a more natural non-master/slave approach has been pursued later, where the cooperative system is seen as a whole. Namely, the reference motion of the object is used to determine the motion of all the arms in the system and the interaction forces acting at each end-effector are fed back so as to be directly controlled. To this aim, the mappings between the forces and velocities at the end-effector of each manipulator and their counterparts at the manipulated object are to be considered in the design of the control laws.

29.6.1 Hybrid Control

In [29.11, 12] a non-master/slave approach has been proposed, based on the well-known scheme proposed by Raibert and Craig for the robot/environment interaction control of single-arm systems (see Chap. 7). The operational space vector for the hybrid position/force control is defined via the following variables

$$x_O = \begin{pmatrix} x_E \\ x_I \end{pmatrix}, \quad (29.65)$$

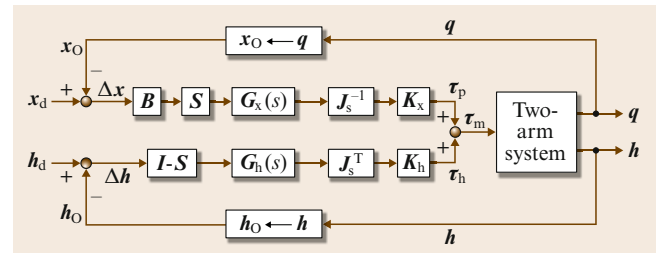


Fig. 29.2 A hybrid position/force control scheme

where $\mathbf{x}_E, \mathbf{x}_I$ are the operational space vectors, defined by specifying the orientation via a minimal set of orientation angles (e.g., Euler angles). The generalized forces vector to be considered is

$$\mathbf{h}_O = \begin{pmatrix} \mathbf{h}_E \\ \mathbf{h}_I \end{pmatrix}. \quad (29.66)$$

The organization of the control scheme is shown diagrammatically in Fig. 29.2. The suffixes d and m represent the desired value and the control command, respectively. The command vector $\boldsymbol{\tau}_m$ to the actuators of the two arms is given by two contributions:

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_p + \boldsymbol{\tau}_h. \quad (29.67)$$

The first term, $\boldsymbol{\tau}_p$, is the command vector for the position control and is given by

$$\boldsymbol{\tau}_p = \mathbf{K}_x \mathbf{J}_s^{-1} \mathbf{G}_x(s) \mathbf{S} \mathbf{B}(\mathbf{x}_{O,d} - \mathbf{x}_O), \quad (29.68)$$

while $\boldsymbol{\tau}_h$ is the command vector for the force control

$$\boldsymbol{\tau}_h = \mathbf{K}_h \mathbf{J}_s^T \mathbf{G}_h(s) (\mathbf{I} - \mathbf{S})(\mathbf{h}_{O,d} - \mathbf{h}_O). \quad (29.69)$$

The matrix \mathbf{B} transforms the errors on the orientation angles into equivalent rotation vectors. The matrix \mathbf{J}_s is the Jacobian matrix that transforms the joint velocity $\dot{\mathbf{q}}$ into the task-space velocity \mathbf{v}_O . The matrix operators $\mathbf{G}_x(s)$ and $\mathbf{G}_h(s)$ represent position and force control laws, respectively. The gain matrices \mathbf{K}_x and \mathbf{K}_h are assumed to be diagonal; their diagonal elements convert velocity and force commands into actuator commands, respectively. The matrix \mathbf{S} selects the position-controlled variables; it is diagonal and its diagonal entries take the values of 1 or 0; namely, the i -th workspace coordinate is position controlled if the i -th diagonal element of \mathbf{S} is 1, while it is force controlled if it is 0. Finally, \mathbf{I} is the identity matrix having the same dimensions as \mathbf{S} , while \mathbf{q} and \mathbf{h} are the vectors of measured joint variables and measured end-effector generalized forces, respectively.

29.6.2 PD Force/Motion Control

In [29.17] a Lyapunov-based approach is pursued to devise force/position PD-type control laws. Namely, the joints torques inputs to each arm are computed as the combination of two contributions:

$$\boldsymbol{\tau}_m = \boldsymbol{\tau}_p + \boldsymbol{\tau}_h, \quad (29.70)$$

where $\boldsymbol{\tau}_p$ is a PD-type term (eventually including a feedback/feedforward model-based compensation term), taking care of position control, while $\boldsymbol{\tau}_h$ is in charge of internal forces/moments control.

Namely, the PD and model-based terms can be computed at the joint level

$$\boldsymbol{\tau}_p = \mathbf{K}_p \mathbf{e}_q - \mathbf{K}_d \dot{\mathbf{q}} + \mathbf{g} + \mathbf{J}^T \mathbf{W}^\dagger \mathbf{g}_E, \quad (29.71)$$

where $\mathbf{e}_q = \mathbf{q}_d - \mathbf{q}$, \mathbf{q}_d is the vector of desired joint variables, \mathbf{K}_p and \mathbf{K}_d are positive-definite matrix gains, \mathbf{g} is the vector of the gravitational forces/torques acting at the manipulators joints, \mathbf{g}_E is the vector of gravity forces/moments at the manipulated object.

Since the cooperative task is usually assigned in terms of absolute and relative motion, the equivalent end-effector desired trajectories are to be computed by using the closed-chain constraints, as in the following example.

Example 29.2 (computation of desired trajectories for a planar two-arm system): For the planar two-arm system, the desired trajectories defining the cooperative task are assigned by specifying the absolute, $\mathbf{x}_{a,d}(t)$, and relative, $\mathbf{x}_{r,d}(t)$, desired motion. Then, the corresponding end-effectors desired trajectories can be computed as

$$\mathbf{x}_{1,d}(t) = \mathbf{x}_{a,d}(t) - \frac{1}{2} \mathbf{x}_{r,d}(t), \quad (29.72)$$

$$\mathbf{x}_{2,d}(t) = \mathbf{x}_{a,d}(t) + \frac{1}{2} \mathbf{x}_{r,d}(t), \quad (29.73)$$

where (29.41) and (29.42) have been exploited.

In the spatial case (29.72, 29.73) do not hold, since the absolute and relative orientation must be expressed in terms of geometrically meaningful quantities, e.g., via (29.44, 29.45). However, the same approach may be pursued at the expense of a slightly more complex expressions of the above formulas [29.26].

Once, the desired position/orientation for each end-effector has been obtained, the inverse kinematics of each arm has to be computed to provide the desired joint trajectories, \mathbf{q}_d , to the control loop; to this aim, e.g., numerical inverse kinematics algorithms may be employed (see Chaps. 1 and 11).

Also, a PD-type control law can be expressed in terms of end-effector variables, i.e.,

$$\boldsymbol{\tau}_p = \mathbf{J}^T (\mathbf{K}_p \mathbf{e} - \mathbf{K}_v \mathbf{v}) - \mathbf{K}_d \dot{\mathbf{q}} + \mathbf{g} + \mathbf{J}^T \mathbf{W}^\dagger \mathbf{g}_E, \quad (29.74)$$

where \mathbf{e} is the tracking error computed in terms of end-effector position/orientation variables, \mathbf{v} is the vector

collecting the end-effector velocities, and \mathbf{K}_p , \mathbf{K}_v , and \mathbf{K}_d are positive-definite matrix gains.

Finally, the same concept may be exploited to design a PD control law directly in the object space

$$\tau_p = \mathbf{J}^\top \mathbf{W}^\dagger (\mathbf{K}_p \mathbf{e}_E - \mathbf{K}_v \mathbf{v}_E) - \mathbf{K}_d \dot{\mathbf{q}} + \mathbf{g} + \mathbf{J}^\top \mathbf{W}^\dagger \mathbf{g}_E, \quad (29.75)$$

where \mathbf{e}_E is the tracking error computed in terms of object absolute position/orientation variables, \mathbf{v}_E is the object's generalized velocity vector, and \mathbf{K}_p , \mathbf{K}_v , and \mathbf{K}_d are positive-definite matrix gains.

The internal force control term can instead be designed as follows

$$\tau_h = \mathbf{J}^\top \mathbf{V} \mathbf{h}_{I,c}, \quad (29.76)$$

where

$$\mathbf{h}_{I,c} = \mathbf{h}_{I,d} + \mathbf{G}_h(s)(\mathbf{h}_{I,d} - \mathbf{h}_I), \quad (29.77)$$

$\mathbf{G}_h(s)$ is a matrix operator representing a strictly proper linear filter, such that $\mathbf{I} - \mathbf{G}_h(s)$ has zeros only in the left half plane, and $\mathbf{h}_{I,d}$ is the vector of desired internal forces; the internal force vector can be computed from the vector of measured end-effector forces as $\mathbf{h}_I = \mathbf{V}^\dagger \mathbf{h}$. A particularly simple choice, ensuring a null error at steady state, is given by

$$\mathbf{G}_h(s) = \frac{1}{s} \mathbf{K}_h,$$

where \mathbf{K}_h is a positive-definite matrix. Remarkably, when an infinitely rigid object/grasp is considered, preprocessing of the force error via a strictly proper filter is needed to ensure closed-loop stability [29.17]; e.g., if a simple proportional feedback is adopted (i.e., $\mathbf{G}_h(s) = \mathbf{K}_h$), the closed loop will be unstable in the presence of an arbitrary small time delay, unless a small (namely, smaller than 1) force gain is adopted. In practice, the closed kinematic chain will be characterized by some elasticity (e.g., due to grippers, end-effector force/torques sensors, joints); in this case, the product between the control gain \mathbf{K}_h and the stiffness of the flexible components has to be chosen sufficiently small to ensure stability.

An interesting extension of the above approach has been given in [29.26, 28], where kinetostatic filtering of the control action is performed so as to filter all the components of the control input which contribute to internal stresses at the object. Namely, the control law (29.71) can be modified by weighting the proportional term ($\mathbf{K}_p \mathbf{e}_E$) via the filtering matrix

$$\phi = \mathbf{J}^\top (\mathbf{W}^\dagger \mathbf{W} + \mathbf{V} \Sigma \mathbf{V}^\dagger) \mathbf{J}^{-\top},$$

where the (6×6) diagonal matrix $\Sigma = \text{diag}\{\sigma_i\}$ weights the components of $\mathbf{J}^{-\top} \mathbf{K}_p \mathbf{e}_E$ in each direction of the subspace of the internal forces via the constant values $0 \leq \sigma_i \leq 1$. In detail, if $\Sigma = \mathbf{O}_6$ then all these components are completely canceled from the control action, while the choice $\Sigma = \mathbf{I}_6$ leads to the control law (29.71) without kinetostatic filtering. In a similar way, the control laws (29.74) and (29.75) can be modified so as to introduce proper kinetostatic filtering of $\mathbf{K}_p \mathbf{e}$ and $\mathbf{K}_p \mathbf{e}_E$, respectively.

29.6.3 Feedback Linearization Approaches

A further improvement of the PD-plus-gravity compensation control approach has been achieved by introducing a full model compensation, so as to achieve feedback/feedforward linearization of the closed-loop system. The feedback linearization approach formulated at the operational space level is the base for the so-called *augmented object* approach [29.63, 64]. In this approach the system is modeled in the operational space as a whole, by suitably expressing its inertial properties via a single augmented inertia matrix \mathbf{M}_O . Hence, the dynamics of the cooperative system in the operational space can be written as

$$\mathbf{M}_O(\mathbf{x}_E) \ddot{\mathbf{x}}_E + \mathbf{c}_O(\mathbf{x}_E, \dot{\mathbf{x}}_E) = \mathbf{h}_E. \quad (29.78)$$

In (29.78), \mathbf{M}_O and \mathbf{c}_O are the operational space terms modeling, respectively, the inertial properties of the whole system (manipulators and object) and the Coriolis, centrifugal, friction, and gravity terms.

In the framework of feedback linearization (formulated in the operational), the problem of controlling the internal forces can be solved, e.g., by resorting to the *virtual linkage* model [29.16] or according to the scheme proposed in [29.29], i.e.,

$$\tau = \mathbf{J}^\top \mathbf{W}^\dagger [\mathbf{M}_O(\ddot{\mathbf{x}}_{E,d} + \mathbf{K}_v \dot{\mathbf{e}}_E + \mathbf{K}_p \mathbf{e}_E) + \mathbf{c}_O] + \mathbf{J}^\top \mathbf{V} \left(\mathbf{h}_{I,d} + \mathbf{K}_h \int (\mathbf{h}_{I,d} - \mathbf{h}_I) dt \right). \quad (29.79)$$

The above control law yields a linear and decoupled closed-loop dynamics

$$\begin{aligned} \ddot{\mathbf{e}}_E + \mathbf{K}_v \dot{\mathbf{e}}_E + \mathbf{K}_p \mathbf{e}_E &= \mathbf{0} \\ \ddot{\mathbf{h}}_I + \mathbf{K}_h \int \ddot{\mathbf{h}}_I dt &= \mathbf{0}, \end{aligned} \quad (29.80)$$

where $\ddot{\mathbf{h}}_I = \mathbf{h}_{I,d} - \mathbf{h}_I$. Hence, the closed-loop dynamics guarantees asymptotically vanishing motion and force errors.

29.6.4 Impedance Control

An alternative control approach can be pursued based on the well-known impedance concept (see Chap. 7). In fact, when a manipulation system interacts with an external environment and/or other manipulators, large values of the contact forces and moments can be avoided by enforcing a compliant behavior, with suitable dynamic features, of the robotic system. Impedance control schemes have been proposed in the case of cooperative manipulation to control object/environment interaction forces [29.21] or internal forces [29.22]. More recently, an impedance scheme for the control of both external forces and internal forces has been proposed [29.65].

In detail, the impedance scheme in [29.21] enforces the following mechanical impedance behavior between the object displacements and the forces acting on object due to interaction with the environment:

$$\mathbf{M}_E \ddot{\mathbf{a}}_E + \mathbf{D}_E \dot{\mathbf{v}}_E + \mathbf{K}_E \mathbf{e}_E = \mathbf{h}_{\text{env}}, \quad (29.81)$$

where:

- \mathbf{e}_E represents the vector of the object's displacements between the desired and actual pose,
- $\dot{\mathbf{v}}_E$ is the difference between the object's desired and actual generalized velocities,
- $\ddot{\mathbf{a}}_E$ is the difference between the object's desired and actual generalized accelerations,

and \mathbf{h}_{env} is the generalized force acting on object due to the interaction with the environment. The impedance dynamics is characterized in terms of given positive-definite mass (\mathbf{M}_E), damping (\mathbf{D}_E), and stiffness (\mathbf{K}_E) matrices to be properly chosen so as to achieve the desired compliant behavior of the object.

As for the pose displacements used in (29.81), special attention has to be paid to the orientation variables.

Example 29.3 (external impedance for a planar two-arm system): For the planar two-arm system the quantities in (29.81) can be defined in a straightforward way. Namely,

$$\mathbf{e}_E = \mathbf{x}_{E,d} - \mathbf{x}_E, \quad (29.82)$$

$$\dot{\mathbf{v}}_E = \dot{\mathbf{x}}_{E,d} - \dot{\mathbf{x}}_E, \quad (29.83)$$

$$\ddot{\mathbf{a}}_E = \ddot{\mathbf{x}}_{E,d} - \ddot{\mathbf{x}}_E, \quad (29.84)$$

where \mathbf{x}_E is the (3×1) vector collecting the object's position and the orientation, while $\mathbf{x}_{E,d}$ represents its desired counterpart. Hence, \mathbf{M}_E , \mathbf{D}_E , and \mathbf{K}_E are (3×3) matrices.

In the spatial case, orientation displacements cannot be defined as in (29.82), and geometrically meaningful

orientation representations (i. e., rotation matrices and/or angle/axis representations) or in terms of operational space variables (i. e., differences between operational space vectors) have to be adopted [29.65].

The impedance scheme in [29.22], enforces a mechanical impedance behavior between the i -th end-effector displacements and the internal forces, i. e.,

$$\mathbf{M}_{I,i} \ddot{\mathbf{a}}_i + \mathbf{D}_{I,i} \dot{\mathbf{v}}_i + \mathbf{K}_{I,i} \mathbf{e}_i = \mathbf{h}_{I,i}, \quad (29.85)$$

where

- \mathbf{e}_i is the vector expressing the displacement of the i -th end-effector between the desired and actual pose,
- $\dot{\mathbf{v}}_i$ is the vector expressing the difference between the desired and actual velocities of the i -th end-effector,
- $\ddot{\mathbf{a}}_i$ is the vector expressing the difference between the desired and actual accelerations of the i -th end-effector,

and $\mathbf{h}_{I,i}$ is the contribution of the i -th end-effector to the internal force, i. e., the i -th component of the vector $\mathbf{V}\mathbf{V}^\dagger \mathbf{h}$. Again, the impedance dynamics is characterized in terms of given positive-definite mass ($\mathbf{M}_{I,i}$), damping ($\mathbf{D}_{I,i}$), and stiffness ($\mathbf{K}_{I,i}$) matrices, to be properly chosen so as to achieve a suitable compliant behavior of the end-effectors with respect to internal forces.

Example 29.4 (Internal impedance for a planar two-arm system): For the planar two-arm system the quantities in (29.85) can be defined in a straightforward way as well

$$\mathbf{e}_i = \mathbf{x}_{i,d} - \mathbf{x}_i, \quad (29.86)$$

$$\dot{\mathbf{v}}_i = \dot{\mathbf{x}}_{i,d} - \dot{\mathbf{x}}_i, \quad (29.87)$$

$$\ddot{\mathbf{a}}_i = \ddot{\mathbf{x}}_{i,d} - \ddot{\mathbf{x}}_i, \quad (29.88)$$

where \mathbf{x}_i is the (3×1) vector collecting the position in the plane and the orientation angle of the i -th end-effector, while $\mathbf{x}_{i,d}$ represents its desired counterpart. Again, $\mathbf{M}_{I,i}$, $\mathbf{D}_{I,i}$, and $\mathbf{K}_{I,i}$ are (3×3) matrices.

As for the spatial case, orientation displacements in (29.85) cannot be defined as in (29.86), and geometrically meaningful orientation representations (i. e., rotation matrices and/or angle/axis representations) or in terms of operational space variables (i. e., differences between operational space vectors) have to be adopted [29.65].

The above two approaches have been combined in [29.65], where two control loops are designed to enforce an impedance behavior both at the object level (external forces) and at the end-effector level (internal forces).

29.7 Conclusions and Further Reading

In this chapter, the fundamentals of cooperative manipulation have been presented. A historical overview of the research on cooperative manipulation has been provided. The kinematics and dynamics of robotic arms cooperatively manipulating a tightly grasped rigid object are considered. Special topics, such as the definition of a cooperative task space and the problem of load distribution, have also been touched on. Then, the main control approaches for cooperative systems have been discussed. A few advanced topics related to control of cooperative robots, i. e., advanced nonlinear control as well as the modeling and control of cooperative systems including flexible elements, will be briefly outlined in the following.

Some of the first attempts to cope with uncertainties and disturbances in cooperative robots control focused on adaptive strategies [29.23, 24], in which the unknown parameters are estimated online, based on a suitable linear-in-the-parameters model of the uncertainties. In detail, the approach in [29.23] is aimed at controlling the object motion, the interaction force due to the contact object/environment, and the internal forces; the adaptive control law estimates the unknown model parameters, both of the manipulators and of the object, on the basis of suitable error equations. In [29.24], the adaptive control concept is used to design a decentralized scheme, i. e., a centralized coordinator is not used, for redundant cooperative manipulators.

A recently developed control framework for cooperative systems is the so-called synchronization control [29.49, 50]; in this approach the control problem is formulated in terms of suitably defined errors accounting for motion synchronization between the manipulators involved in the cooperative task. Namely, the key idea in [29.49] is to ensure tracking of the desired trajectory assigned to each manipulator, while synchronizing its motion with other manipulators motion. In [29.50], the problem of synchronizing the motion of multiple manipulators systems is solved by using only position measurements; the synchronization controller consists of a feedback control law and a set of nonlinear observers, and synchronization is ensured by suitably defining the coupling errors between the manipulators' motions.

Recently, research efforts have been spent on intelligent control [29.51, 52]. Namely, in [29.51], a semi-decentralized adaptive fuzzy control scheme, with \mathcal{H}_∞ performance in motion and internal force

tracking, is proposed; the controller of each robot consists of two parts: a model-based adaptive controller and an adaptive fuzzy logic controller; the model-based adaptive controller handles the nominal dynamics, including a purely parametric uncertainties model, while the fuzzy logic controller is aimed at counteracting the effect of unstructured uncertainties and external disturbances. In [29.52], a decentralized adaptive fuzzy control scheme is proposed, where the control law makes use of a multi-input multi-output fuzzy logic engine and a systematic online adaptation mechanism.

Finally it is worth mentioning the efforts invested in the investigation of control strategies using partial state feedback, i. e., only joints position and end-effector forces are fed back to the controller. A recent contribution was provided by the work in [29.53], in which a decentralized control algorithm that achieves asymptotic tracking of desired positions and forces by using a nonlinear observer for velocities was proposed.

Flexibility in a cooperative system may arise, e.g., due to the use of nonrigid grippers to grasp the manipulated object. In fact, the adoption of compliant grippers allows large internal forces to be avoided and, at the same time, achieves safe manipulation of the object, even in the presence of failures and unpredicted contacts with the external environment. In detail, the work in [29.57] develops a non-model-based decentralized control scheme. Namely, a proportional-derivative (PD) position feedback control scheme with gravity compensation, is designed; the PD scheme is capable of regulating the position/orientation of the manipulated object and, simultaneously, achieves damping of the vibrations induced by the compliant grippers; also, a hybrid scheme is adopted to control internal forces along the directions in which the compliance of grippers is too low to ensure limited internal stresses at the manipulated object.

Other research efforts have been focused on handling multibodied objects, or even flexible objects [29.42–44]. Those objects are difficult to handle and, therefore, assembly of those objects in manufacturing industry is not automated. Also, cooperative control of multi-flexible-arm robots has been investigated [29.45, 46]. Once the modeling and control problem is solved (see Chap. 13), the flexible-arm robot is a robot with many merits [29.47]: it is lightweight, compliant, and hence

safe, etc. Combining control methods for flexible-arm robots, such as vibration suppression, with the co-operative control methods presented in this chapter

is straightforward [29.45]. Automated object-retrieval operation with a two-flexible-arm robot has been demonstrated in [29.46].

References

- 29.1 S. Fujii, S. Kurono: Coordinated computer control of a pair of manipulators, *Proc. 4th IFTOMM World Congress (Newcastle upon Tyne 1975)* pp. 411–417
- 29.2 E. Nakano, S. Ozaki, T. Ishida, I. Kato: Cooperational control of the anthropomorphous manipulator 'MELARM', *Proc. 4th Int. Symp. Ind. Robots (Tokyo 1974)* pp. 251–260
- 29.3 K. Takase, H. Inoue, K. Sato, S. Hagiwara: The design of an articulated manipulator with torque control ability, *Proc. 4th Int. Symp. Ind. Robots (Tokyo 1974)* pp. 261–270
- 29.4 S. Kurono: Cooperative control of two artificial hands by a mini-computer, *Prepr. 15th Joint Conf. on Automatic Control (1972)* pp. 365–366, (in Japanese)
- 29.5 A.J. Koivo, G.A. Bekey: Report of workshop on co-ordinated multiple robot manipulators: planning, control, and applications, *IEEE J. Robot. Autom.* **4**(1), 91–93 (1988)
- 29.6 P. Dauchez, R. Zapata: Co-ordinated control of two cooperative manipulators: the use of a kinematic model, *Proc. 15th Int. Symp. Ind. Robots (Tokyo 1985)* pp. 641–648
- 29.7 N.H. McClamroch: Singular systems of differential equations as dynamic models for constrained robot systems, *Proc. 1986 IEEE Int. Conf. on Robotics and Automation (San Francisco 1986)* pp. 21–28
- 29.8 T.J. Tarn, A.K. Bejczy, X. Yun: New nonlinear control algorithms for multiple robot arms, *IEEE Trans. Aerosp. Electron. Syst.* **24**(5), 571–583 (1988)
- 29.9 S. Hayati: Hybrid position/force control of multi-arm cooperating robots, *Proc. 1986 IEEE Int. Conf. on Robotics and Automation (San Francisco 1986)* pp. 82–89
- 29.10 M. Uchiyama, N. Iwasawa, K. Hakomori: Hybrid position/force control for coordination of a two-arm robot, *Proc. 1987 IEEE Int. Conf. on Robotics and Automation (Raleigh 1987)* pp. 1242–1247
- 29.11 M. Uchiyama, P. Dauchez: A symmetric hybrid position/force control scheme for the coordination of two robots, *Proc. 1988 IEEE Int. Conf. on Robotics and Automation (Philadelphia 1988)* pp. 350–356
- 29.12 M. Uchiyama, P. Dauchez: Symmetric kinematic formulation and non-master/slave coordinated control of two-arm robots, *Adv. Robot.* **7**(4), 361–383 (1993)
- 29.13 I.D. Walker, R.A. Freeman, S.I. Marcus: Analysis of motion and internal force loading of objects grasped by multiple cooperating manipulators, *Int. J. Robot. Res.* **10**(4), 396–409 (1991)
- 29.14 R.G. Bonitz, T.C. Hsia: Force decomposition in co-operating manipulators using the theory of metric spaces and generalized inverses, *Proc. 1994 IEEE Int. Conf. on Robotics and Automation (San Diego 1994)* pp. 1521–1527
- 29.15 D. Williams, O. Khatib: The virtual linkage: a model for internal forces in multi-grasp manipulation, *Proc. 1993 IEEE Int. Conf. on Robotics and Automation (Atlanta 1993)* pp. 1025–1030
- 29.16 K.S. Sang, R. Holmberg, O. Khatib: The augmented object model: cooperative manipulation and parallel mechanisms dynamics, *Proceedings of the 2000 IEEE International Conference on Robotics and Automation (San Francisco 1995)* pp. 470–475
- 29.17 J.T. Wen, K. Kreutz-Delgado: Motion and force control of multiple robotic manipulators, *Automatica* **28**(4), 729–743 (1992)
- 29.18 T. Yoshikawa, X.Z. Zheng: Coordinated dynamic hybrid position/force control for multiple robot manipulators handling one constrained object, *Int. J. Robot. Res.* **12**, 219–230 (1993)
- 29.19 V. Perdureau, M. Drouin: Hybrid external control for two robot coordinated motion, *Robotica* **14**, 141–153 (1996)
- 29.20 H. Bruhm, J. Deisenroth, P. Schadler: On the design and simulation-based validation of an active compliance law for multi-arm robots, *Robot. Auton. Syst.* **5**, 307–321 (1989)
- 29.21 S.A. Schneider, R.H. Cannon Jr.: Object impedance control for cooperative manipulation: Theory and experimental results, *IEEE Trans. Robot. Autom.* **8**, 383–394 (1992)
- 29.22 R.G. Bonitz, T.C. Hsia: Internal force-based impedance control for cooperating manipulators, *IEEE Trans. Robot. Autom.* **12**, 78–89 (1996)
- 29.23 Y.-R. Hu, A.A. Goldenberg, C. Zhou: Motion and force control of coordinated robots during constrained motion tasks, *Int. J. Robot. Res.* **14**, 351–365 (1995)
- 29.24 Y.-H. Liu, S. Arimoto: Decentralized adaptive and nonadaptive position/force controllers for redundant manipulators in cooperation, *Int. J. Robot. Res.* **17**, 232–247 (1998)
- 29.25 P. Chiacchio, S. Chiaverini, B. Siciliano: Direct and inverse kinematics for coordinated motion tasks of a two-manipulator system, *ASME J. Dyn. Syst. Meas. Contr.* **118**, 691–697 (1996)
- 29.26 F. Caccavale, P. Chiacchio, S. Chiaverini: Task-Space regulation of cooperative manipulators, *Automatica* **36**, 879–887 (2000)

- 29.27 G.R. Luecke, K.W. Lai: A joint error-feedback approach to internal force regulation in cooperating manipulator systems, *J. Robot. Syst.* **14**, 631–648 (1997)
- 29.28 F. Caccavale, P. Chiacchio, S. Chiaverini: Stability analysis of a joint space control law for a two-manipulator system, *IEEE Trans. Autom. Contr.* **44**, 85–88 (1999)
- 29.29 P. Hsu: Coordinated control of multiple manipulator systems, *IEEE Trans. Robot. Autom.* **9**, 400–410 (1993)
- 29.30 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Global task space manipulability ellipsoids for multiple arm systems, *IEEE Trans. Robot. Autom.* **7**, 678–685 (1991)
- 29.31 S. Lee: Dual redundant arm configuration optimization with task-oriented dual arm manipulability, *IEEE Trans. Robot. Autom.* **5**, 78–97 (1989)
- 29.32 T. Kokkinis, B. Paden: Kinetostatic performance limits of cooperating robot manipulators using force-velocity polytopes, *Proc. of ASME Winter Annual Meeting—robotics Research* (San Francisco 1989)
- 29.33 P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano: Task space dynamic analysis of multiarm system configurations, *Int. J. Robot. Res.* **10**, 708–715 (1991)
- 29.34 D.E. Orin, S.Y. Oh: Control of force distribution in robotic mechanisms containing closed kinematic chains, *Trans. ASME J. Dyn. Syst. Meas. Contr.* **102**, 134–141 (1981)
- 29.35 Y.F. Zheng, J.Y.S. Luh: Optimal load distribution for two industrial robots handling a single object, *Proc. 1988 IEEE Int. Conf. on Robotics and Automation* (Philadelphia 1988) pp. 344–349
- 29.36 I.D. Walker, S.I. Marcus, R.A. Freeman: Distribution of dynamic loads for multiple cooperating robot manipulators, *J. Robot. Syst.* **6**, 35–47 (1989)
- 29.37 M. Uchiyama: A unified approach to load sharing, motion decomposing, and force sensing of dual arm robots, *Robotics Research: 5th Int. Symp.*, ed. by H. Miura, S. Arimoto (MIT, 1990) pp. 225–232
- 29.38 M.A. Unseren: A new technique for dynamic load distribution when two manipulators mutually lift a rigid object. Part 1: The proposed technique, *Proc. First World Automation Congress (WAC '94)*, Vol. 2 (Maui 1994) pp. 359–365
- 29.39 M.A. Unseren: A new technique for dynamic load distribution when two manipulators mutually lift a rigid object. Part 2: Derivation of entire system model and control architecture, *Proc. First World Automation Congress (WAC '94)*, Vol. 2 (Maui 1994) pp. 367–372
- 29.40 M. Uchiyama, T. Yamashita: Adaptive load sharing for hybrid controlled two cooperative manipulators, *Proc. 1991 IEEE Int. Conf. on Robotics and Automation* (Sacramento 1991) pp. 986–991
- 29.41 M. Uchiyama, Y. Kanamori: Quadratic programming for dextrous dual-arm manipulation. In: *Robotics, Mechatronics and Manufacturing Systems, Trans. IMACS/SICE Int. Symp. on Robotics, Mechatronics and Manufacturing Systems, Kobe, Japan, September 1992*, ed. by T. Takamori, K. Tsuchiya (Elsevier, North-Holland 1993) pp. 367–372
- 29.42 Y.F. Zheng, M.Z. Chen: Trajectory planning for two manipulators to deform flexible beams, *Proc. 1993 IEEE Int. Conf. on Robotics and Automation* (Atlanta 1993) pp. 1019–1024
- 29.43 M.M. Svinin, M. Uchiyama: Coordinated dynamic control of a system of manipulators coupled via a flexible object, *Prepr. 4th IFAC Symp. on Robot Control* (Capri 1994) pp. 1005–1010
- 29.44 T. Yukawa, M. Uchiyama, D.N. Nenchev, H. Inooka: Stability of control system in handling of a flexible object by rigid arm robots, *Proc. 1996 IEEE Int. Conf. on Robotics and Automation* (Minneapolis 1996) pp. 2332–2339
- 29.45 M. Yamano, J.-S. Kim, A. Konno, M. Uchiyama: Cooperative control of a 3D dual-flexible-arm robot, *J. Intell. Robot. Syst.* **39**, 1–15 (2004)
- 29.46 T. Miyabe, A. Konno, M. Uchiyama, M. Yamano: An approach toward an automated object retrieval operation with a two-arm flexible manipulator, *Int. J. Robot. Res.* **23**, 275–291 (2004)
- 29.47 M. Uchiyama, A. Konno: Modeling, controllability and vibration suppression of 3D flexible robots. In: *Robotics Research, The 7th Int. Symp.*, ed. by G. Giralt, G. Hirzinger (Springer, London 1996) pp. 90–99
- 29.48 K. Munawar, M. Uchiyama: Slip compensated manipulation with cooperating multiple robots, *36th IEEE CDC* (San Diego 1997)
- 29.49 D. Sun, J.K. Mills: Adaptive synchronized control for coordination of multirobot assembly tasks, *IEEE Trans. Robot. Autom.* **18**, 498–510 (2002)
- 29.50 A. Rodriguez-Angeles, H. Nijmeijer: Mutual synchronization of robots via estimated state feedback: a cooperative approach, *IEEE Trans. Contr. Syst. Technol.* **12**, 542–554 (2004)
- 29.51 K.-Y. Lian, C.-S. Chiu, P. Liu: Semi-decentralized adaptive fuzzy control for cooperative multirobot systems with H-inf motion/internal force tracking performance, *IEEE Trans. Syst. Man Cybern. – Part B: Cybernetics* **32**, 269–280 (2002)
- 29.52 W. Gueaieb, F. Karray, S. Al-Sharhan: A robust adaptive fuzzy position/force control scheme for cooperative manipulators, *IEEE Trans. Contr. Syst. Technol.* **11**, 516–528 (2003)
- 29.53 J. Gudiño-Lau, M.A. Arteaga, L.A. Muñoz, V. Parra-Vega: On the control of cooperative robots without velocity measurements, *IEEE Trans. Contr. Syst. Technol.* **12**, 600–608 (2004)
- 29.54 H. Inoue: Computer controlled bilateral manipulator, *Bull. JSME* **14**(69), 199–207 (1971)

- 29.55 M. Uchiyama, T. Kitano, Y. Tanno, K. Miyawaki: Cooperative multiple robots to be applied to industries, Proc. World Automation Congress (WAC '96), Vol. 3 (Montpellier 1996) pp. 759–764
- 29.56 B.M. Braun, G.P. Starr, J.E. Wood, R. Lumia: A framework for implementing cooperative motion on industrial controllers, IEEE Trans. Robot. Autom. **20**, 583–589 (2004)
- 29.57 D. Sun, J.K. Mills: Manipulating rigid payloads with multiple robots using compliant grippers, IEEE/ASME Trans. Mechatron. **7**, 23–34 (2002)
- 29.58 J.Y.S. Luh, Y.F. Zheng: Constrained relations between two coordinated industrial robots for motion control, Int. J. Robot. Res. **6**, 60–70 (1987)
- 29.59 A.J. Koivo, M.A. Unseren: Reduced order model and decoupled control architecture for two manipulators holding a rigid object, ASME J. Dyn. Syst. Meas. Contr. **113**, 646–654 (1991)
- 29.60 M.A. Unseren: Rigid body dynamics and decoupled control architecture for two strongly interacting manipulators manipulators, Robotica **9**, 421–430 (1991)
- 29.61 J. Duffy: The fallacy of modern hybrid control theory that is based on “Orthogonal Complements” of twist and wrench spaces, J. Robot. Syst. **7**, 139–144 (1990)
- 29.62 K.L. Doty, C. Melchiorri, C. Bonivento: A theory of generalized inverses applied to robotics, Int. J. Robot. Res. **12**, 1–19 (1993)
- 29.63 O. Khatib: Object manipulation in a multi-effector robot system,. In: *Robotics Research*, Vol. 4, ed. by R. Bolles, B. Roth (MIT Press, Cambridge 1988) pp. 137–144
- 29.64 O. Khatib: Inertial properties in robotic manipulation: An object level framework, Int. J. Robot. Res. **13**, 19–36 (1995)
- 29.65 F. Caccavale, L. Villani: Impedance control of co-operative manipulators, Mach. Intell. Robot. Contr. **2**, 51–57 (2000)

Haptics

30. Haptics

Blake Hannaford, Allison M. Okamura

The word *haptics*, believed to be derived from the Greek word *haptesthai*, means *related to the sense of touch*. In the psychology and neuroscience literature, haptics is the study of human touch sensing, specifically via *kinesthetic* (force/position) and *cutaneous* (tactile) receptors, associated with perception and manipulation. In the robotics and virtual reality literature, haptics is broadly defined as real and simulated touch interactions between robots, humans, and real, remote, or simulated environments, in various combinations. This chapter focuses on the use of specialized robotic devices and their corresponding control, known as *haptic interfaces*, that allow human operators to experience the sense of touch in remote (teleoperated) or simulated (virtual) environments.

30.1 Overview 719

30.1.1 Human Haptics 721

30.1.2 Application Examples 722

30.2 Haptic Device Design 724

30.2.1 Mechanisms 724

30.2.2 Sensing 725

30.2.3 Actuation and Transmission 726

30.2.4 An Example Device 726

30.3 Haptic Rendering 727

30.3.1 Rendering Complex Environments . 729

30.3.2 Virtual Coupling 729

30.4 Control and Stability of Haptic Interfaces 730

30.5 Tactile Display 731

30.5.1 Vibration Feedback 731

30.5.2 Contact Location, Slip, and Shear Display 732

30.5.3 Local Shape 733

30.5.4 Temperature 734

30.6 Conclusions and Further Reading 735

References 735

Haptic technology is intimately connected with robotics through its reliance on dexterous mechatronic devices and draws heavily on the theoretical foundations of manipulator design, actuation, sensing, and control. In this chapter, we begin with motivation for haptic interface design and use, including the basic design of a haptic interface, information about human haptics, and examples of haptic interface applications. Next, we review concepts in the mechatronic design of kines-

thetic haptic interfaces, including sensors, actuators, and mechanisms. We then examine the control aspect of kinesthetic haptic interfaces, particularly the rendering of virtual environments and stable and accurate display of forces. We next review tactile displays, which vary widely in their design due to the many types of tactile information that can be presented to a human operator. Finally, we provide resources for further study of haptics.

30.1 Overview

Haptics is the science and technology of experiencing and creating touch sensations in human operators. Imagine trying to button a coat, shake someone’s hand, or

write a note without the sense of touch. These simple tasks become extremely difficult to perform without adequate haptic feedback. To improve human operator

performance in simulated and teleoperated environments, haptic interfaces seek to generate a compelling sensation that the operator is directly touching a real environment.

Haptic interfaces attempt to replicate or enhance the touch experience of manipulating or perceiving a real environment through mechatronic devices and computer control. They consist of a haptic device (a manipulator with sensors and actuators) and a control computer with software relating human operator inputs to haptic

information display. While the low-level design of haptic interfaces varies widely depending on the application, their operation generally follows the *haptic loop* shown in Fig. 30.1. First, the haptic device *senses* an operator input, which may be position (and its derivatives), force, muscle activity, etc. Second, the sensed input is applied to a virtual or teleoperated environment. For a virtual environment, the effect of the operator's input on virtual objects and the subsequent response to be displayed to the operator are computed based on models and a *haptic rendering* algorithm. In teleoperation, a manipulator that is remote in space, scale, or power attempts to track the operator's input. When the manipulator interacts with its real environment, haptic information to be relayed to the operators is recorded or estimated. Finally, *actuators* on the haptic device are used to physically convey touch sensations to the human operator. Based on the haptic feedback, whether through unconscious or conscious human control, or simply system dynamics, the operator input is modified. This begins another cycle of the haptic loop.

Despite the simplicity of the concept of haptic display, there exist many challenges to developing compelling haptic interfaces. Many of these are addressed through fundamental robotics theory and an understanding of human haptic capabilities. In general, haptic interface performance specifications are based on human sensing and motor control characteristics. One major challenge in artificially generating haptic sensations is that the human operator's motion should be unrestricted when there is no contact with a virtual or remote object. Haptic devices must allow the human operator to make desired motions, thus requiring back-drivability and sufficient degrees of freedom of motion. A variety of robotic designs are used in haptic devices, including exoskeletons, actuated grippers, parallel and serial manipulators, small-workspace *mouse*-like devices, and large-workspace devices that capture whole arm, and even whole body, movement. Another challenge is that humans integrate kinesthetic (force/position) and cutaneous (tactile) information with motion and control cues to form haptic perceptions. Haptic devices would ideally include both force and tactile displays, although this has been rarely done due to size and weight limitations of actuators. Because of the human's sensitivity to high-frequency information, for many haptic interfaces and applications, this loop must repeat at a high frequency, typically 1 kHz. Not only does a high update rate provide realistic (nondiscretized) touch sensations to the human operator, it also typically helps to maintain system stability. Controls analysis for haptic devices must consider

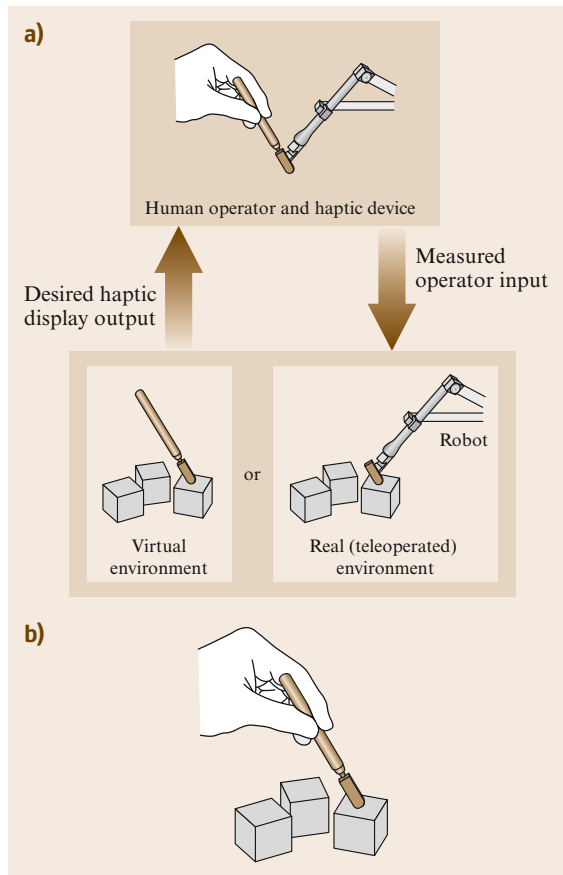


Fig. 30.1 (a) The *haptic loop* of a generic haptic interface. A haptic device senses human operator input, such as position or force, and the system applies this input to a virtual or teleoperated environment. The response of the environment to be relayed to the human operator is computed through models, haptic rendering, sensing, and/or estimation. Finally, actuators on the haptic device display corresponding touch sensations to the human operator. (b) The ideal result is that the human operator feels that he or she is interacting directly with a real environment

both the continuous nature of the physical dynamics and the discrete nature of the computer control.

Before we examine the various components of haptic interfaces in detail, it is useful to motivate their design through a review of human haptics and applications of haptics. The remainder of this section is devoted to those topics.

30.1.1 Human Haptics

Anatomy and Physiology

Two functions of the human nervous system play a primary role in haptics: kinesthesia (the internal sensing of forces and displacements inside muscles, tendons, and joints), and tactile sensing (the sensation of deformations of the skin). Haptics incorporates both, and is associated with an activity such as manipulation or exploration. Most of this chapter will address systems meant to interact primarily with the kinesthetic modality. Devices specifically aimed at tactile perception are described in Sect. 30.5. Even if tactile stimuli are not explicitly generated by a haptic device, tactile receptors are still stimulated, and are known to respond to frequencies as high as 10 000 Hz [30.2] and displacements as small as 2–4 μm [30.3–5].

Kinesthesia is mediated by muscle spindles, which transduce stretch of muscles, and Golgi tendon organs, which transduce joint rotation, especially at the extremes of motion. In principle, these and similar receptors could be stimulated directly to produce haptic sensations. For example, a vibration applied to a muscle tendon creates a strong sensation of muscle lengthening and corresponding joint motion in humans [30.6, 7]. Research in peripheral nerve stimulation for prosthesis control has demonstrated that electrodes implanted within individual fascicles of peripheral nerve stumps in amputees can be stimulated to produce sensations of touch or movement referred to the amputee's phantom hand [30.8].

Psychophysics

At the next level up from physiology and anatomy, psychophysics [30.1], the science of the physical capabilities of the senses, has been a rich source of design data for haptic device development. Its chief contribution has been methodologies that haptics researchers have applied to answer questions about what capabilities are needed in haptic devices. These sensory capabilities could then be translated into design requirements. Some of the chief psychophysical methods that have been fruitfully applied to haptics include threshold mea-

surement by the method of limits and adaptive up-down methods. However, perception at threshold is not 100% reliable. Perception accuracy tends to depend on the strength of the stimulus, and the tradeoff between hit rate and false alarms depends strongly on the probability that a stimulus will be present in a given time interval (P_{stim}).

A more general notion is the receiver operating curve (Fig. 30.2, borrowed by psychophysics from radar theory), which plots the probability of a subject response given the existence of a stimulus, versus the probability of response given no stimulus. The curve is generated by measuring both probabilities at several different values of P_{stim} . The ideal response is the point (0,1): 100% response for stimuli and 0% response for nonstimuli. Human response is near this point for stimuli much above threshold, but declines to a rounded curve and eventually the 45° line as response below threshold becomes equal to chance.

Another relevant concept from psychophysics is the just noticeable difference (JND), commonly expressed as a percentage. This is the magnitude of a relative change in a stimulus, such as a force or displacement applied to the finger that is just perceivable by subjects. For example, Jones [30.9] has measured JNDs of 6%

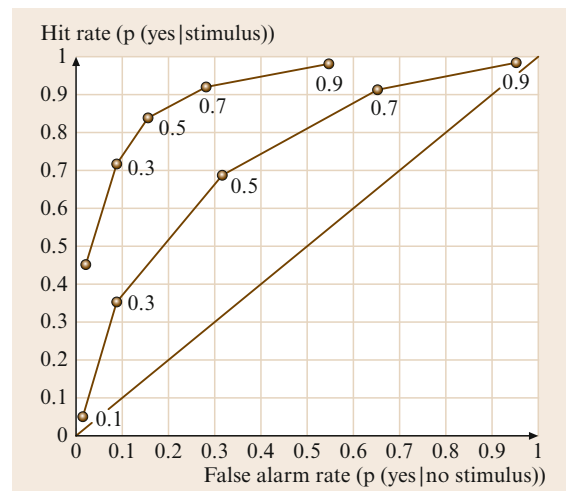


Fig. 30.2 Receiver operating curve (ROC). The ROC encodes the tradeoffs made by a subject between risk of false alarm and risk of missing a valid stimulus. Each point encodes a specific tradeoff between these two risks observed when stimuli are presented with a specified probability. ROCs for stronger signals tend towards the upper left hand corner. (after [30.1] with permission from Lawrence Erlbaum and Associates, Mahwah)

for force applied to the human finger over a range of 0.5–200 N.

Psychology: Exploratory Procedures

In influential research starting in the 1980s, *Lederman* and *Klatzky* defined stereotyped hand motions called *exploratory procedures* (EPs), which are characteristic of human haptic exploration [30.10–12]. They placed objects into the hands of blindfolded subjects, and videotaped their hand motions. Their initial experiments [30.10] showed that the EPs used by subjects could be predicted based on the object property (texture, mass, temperature, etc.) that the subjects needed to discriminate. They also showed that the EPs chosen by subjects were the ones best able to discriminate that property. Furthermore, when asked to answer specific questions about objects (*Is this eating utensil further a fork?*), subjects used a two-stage sequence in which a more general lifting EP preceded more specific EPs [30.12].

Lederman and Klatzky's eight EPs (Fig. 30.3) and the property for which they are optimal are:

1. lateral motion (texture)
2. pressure (hardness)
3. static contact (temperature)
4. unsupported holding (weight)
5. enclosure (global shape, volume)
6. contour following (exact shape, volume)
7. part motion test (part motion)
8. function testing (specific function)

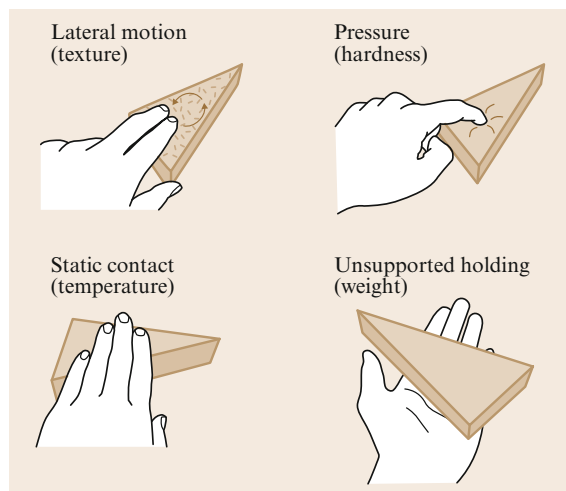


Fig. 30.3 Four of the eight human exploratory procedures (EPs) (after *Lederman* and *Klatzky* [30.10])

Each of these EPs is a bimanual task involving contact with interior surfaces of the hand, motion of the wrist, various degrees of freedom of the, and tactile and temperature sensors in the skin (e.g., EPs 1 and 3), and kinesthetic sensors in the arm (EP 4). A haptic device capable of supporting all of these EPs is far beyond today's state of the art. However, the significance of these results for the design of haptic interface is great, since they allow us to derive device requirements from EPs.

30.1.2 Application Examples

The most common haptic device encountered by the general population is a vibration display device that provides haptic feedback while an operator plays a video game. For example, when the operator drives off the virtual road or bumps into a virtual wall, the hand controller shakes to imply driving over a rough surface or displays an impulse to represent the shock of hitting a hard surface. We examine two more pragmatic examples, medical simulators and computer-aided design systems, in detail below. In addition, we review several commercially available haptic devices. Although haptic interfaces are not yet in widespread commercial use outside of entertainment, they are being integrated into numerous applications where the potential benefits are clear enough to justify the adoption of new technologies. A variety of novel and creative applications are being developed regularly in numerous fields, including

- assistive technology
- automotive
- design
- education
- entertainment
- human–computer interaction
- manufacturing/assembly
- medical simulation
- micro/nanotechnology
- molecular biology
- prosthetics
- rehabilitation
- scientific visualization
- space
- surgical robotics

Medical Simulations

A major example driving much of today's haptic virtual environment research is simulation for training of

hands-on medical procedures. Medical invasive therapeutic and diagnostic procedures, ranging from drawing blood samples to surgery, are potentially dangerous and painful for the patient and require the student to learn hands-on skills mediated by haptic information pathways [30.13]. Simulators both with and without haptic feedback aim to replace supervised learning directly on human patients or on animals. Simulators have proven highly effective in developing minimally invasive surgery skills [30.14], especially when haptic feedback is provided in early training [30.15]. Expected benefits of training with haptic simulators include

- reduced risk to patients both during and immediately after training
- increased ability to simulate unusual conditions or medical emergencies
- ability to collect physical data during the training process and provide specific and directed feedback to the student
- increased training time per unit of instructor effort

Approaches for simulator designs, specific medical applications, and training evaluation methods have also been widely studied in the last two decades, e.g., [30.16, 17]. However, the costs of this technology are still high. In addition, it is not always clear which improvements in simulator technology, such as haptic device performance or accuracy of soft-tissue modeling, lead to improved clinical performance and, ultimately, patient outcomes.

Computer-Aided Design

The Boeing Company [30.18] has studied the use of haptic interfaces for solving advanced problems in computer-aided design (CAD). One such problem is verification of the ability to efficiently maintain a complex system such as an aircraft. In the past, mechanics could verify the procedures (such as change-out of parts) on a physical prototype. However, this analysis is difficult or impossible to perform visually on an advanced CAD system. The VoxMap Pointshell system (Fig. 30.4) was developed to allow test extraction of parts with a haptic interface. Force sensations from the haptic interface reproduce for the operator the physical constraints of the part bumping into elements of the complex workplace. If the operator can remove the part in the haptic interface, it is verified that this part can be maintained without undue disassembly of the aircraft. This capability has been proved useful in actual design activities.

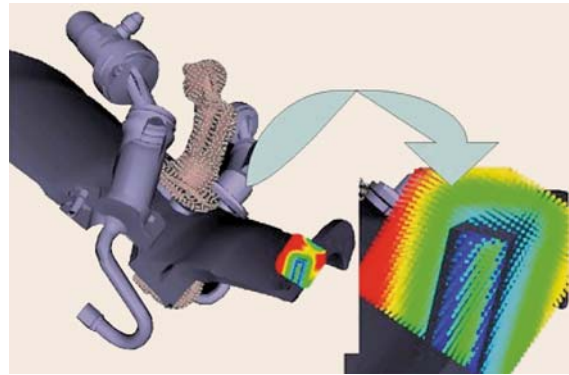


Fig. 30.4 Boeing computer-aided design (CAD) application for assembly and maintenance verification of complex aircraft systems. Boeing researchers developed the Voxmap/Pointshell software for haptically rendering very complex models in six degrees of freedom at high rates. (Courtesy of Bill McNeely, Boeing Phantom Works)

Commercially Available Haptic Devices and Systems

There are a wide variety of haptic devices available for purchase, ranging from high-fidelity research devices to inexpensive entertainment systems. Some researchers also build their own haptic devices in order to implement novel designs, meet particular application constraints, or save costs. At the time of this writing, the bulk of commercially available haptic devices are designed by two companies: SensAble Technologies [30.19] and Immersion Corporation [30.20]. SensAble has developed the Phantom line of stylus-type haptic devices



Fig. 30.5 The Phantom Omni device from SensAble Technologies. This low-cost device senses motion in six degrees of freedom from the stylus and can apply forces in the x , y , and z directions to the stylus tip. (Courtesy SensAble Technologies, Inc., Woburn)

(Fig. 30.5). The Phantom Premium [30.21] has been the most widely used haptic device in research to date. The high price of haptic devices (compared to visual displays) restricts the development of some commercial applications. The Phantom Omni, which is an order of magnitude less expensive than the Phantom Premium, has also gained popularity among haptics and robotics researchers. In 2007, Novint Technologies [30.22] released the Novint Falcon, an inexpensive three-degree-of-freedom haptic device that is in turn an order of magnitude less expensive than the Phantom Omni. This device is aimed more toward entertainment applications.

Immersion has aimed at the mass market and consumer segments with a wide variety of haptics-based products, many of them involving a single degree of

freedom. For example, they have licensed technology to makers of various video games, as well as mobile phone manufacturers, in the form of vibratory feedback in handheld devices and haptic-enabled steering wheels for driving games. Immersion also has a medical division selling medical simulators with haptic feedback.

Software for haptic rendering has also become widely available, through both commercial sources and research groups. Most companies that sell haptic devices also provide a standard development kit (SDK) with haptic rendering capability. In addition, not-for-profit open-source projects such as Chai3D (www.chai3d.org) aim to make rendering algorithms from different groups publicly available, shortening application development time and allowing direct comparison of algorithms for benchmarking purposes.

30.2 Haptic Device Design

There are two broad classes of haptic devices: *admittance* and *impedance* devices. Admittance devices sense the force applied by the operator and constrain the operator's position to match the appropriate deflection of a simulated object or surface in a virtual world. In contrast, an impedance haptic device senses the position of the operator, and then applies a force vector to the operator according to computed behavior of the simulated object or surface.

Robots of the impedance type are back-drivable, have low friction and inertia, and have force-source actuators. A commonly used impedance haptic device in robotics-related research is the Phantom Premium [30.21, 23]. Robots of the admittance type, such as typical industrial robots, are non-back-drivable and have velocity-source actuators. The velocity is controlled with a high-bandwidth low-level controller, and is assumed to be independent of applied external forces. Some commercially available haptic devices, such as the Haptic-Master [30.24], do operate under admittance control. While such closed-loop force control has been used for haptic display, more commonly designers have opted for mechanisms specially designed for open-loop force control to achieve simultaneously low cost and high bandwidth.

The choice of admittance or impedance architecture has many profound implications in the design of the software and hardware system. For a variety of reasons, including cost, the majority of haptic devices implemented today are of the impedance type. Because the preponderance of systems today are impedance de-

vices and limitations on space, we limit our subsequent discussion to that class.

30.2.1 Mechanisms

Creating high-fidelity haptic sensations in the operator requires attention to mechanism design (Chap. 4). The requirements for impedance haptic devices are similar to those for designing manipulators suitable for force control (Chap. 8). Desirable mechanism attributes for open-loop force control include low inertia, high stiffness, and good kinematic conditioning throughout a workspace designed to effectively match the appropriate human limb, primarily the finger or arm. The weight of the mechanism should be minimized, as it is perceived by the operator as weight and inertia of the virtual or teleoperated environment. Kinematic singularities (Chaps. 1, 4, and 12) are detrimental to haptic interfaces because they create directions in space in which the end-point cannot be moved by the human operator and thus impose disturbances on the illusion of haptic contact with virtual objects. High transmission ratios must be avoided as they introduce significant amounts of friction. This constraint requires haptic interfaces to make high demands on actuator performance.

Measures of Mechanism Performance

The ideal haptic device can move freely in any direction and is free of singular configurations as well as the *bad* effects of operating in their neighborhood. Traditionally, kinematic performance has been derived from

the mechanism's Jacobian matrix, $\mathbf{J}(p, q)$, using some of the following well-known measures:

- manipulability [30.25]: the product of the singular values of $\mathbf{J}(p, q)$
- mechanism isotropy [30.26]: the ratio of smallest to the largest singular value of $\mathbf{J}(p, q)$
- minimum force output [30.25, 27, 28]: maximizing the force output in the *worst* direction

Dynamics can also be introduced into the cost function using measures such as dynamic manipulability [30.29]. This is still an active area of research and there is no consensus yet on which dexterity measure is most appropriate for haptic devices.

Kinematic and Dynamic Optimization

This aspect of design requires synthesis of mechanisms that match the workspace of the (most often) human finger or arm while simultaneously avoiding kinematic singularities.

A haptic device workspace is defined to match that of the targeted human limb. This can be assisted by the use of anthropometric data [30.30]. The performance goals, such as low inertia and avoidance of kinematic singularities, must be formalized into a quantitative performance measure which can be computed for any candidate design. Such a measure must account for

- uniformity of kinematic conditioning throughout the target workspace
- favoring designs with lower inertia
- guaranteeing that the target workspace is reachable

The measures of mechanism performance defined above operate at a single point in space and thus must be integrated over the entire workspace to derive a figure of merit for a proposed haptic device design. For example, if S is the set of all joint angles Θ , such that the end effector is inside the target workspace, one such measure is

$$M = \min_S W(\Theta), \quad (30.1)$$

where $W(\Theta)$ is a measure of design performance. The performance measure should include a link length penalty such as

$$M = \min_S \frac{W(\Theta)}{l^3}, \quad (30.2)$$

in order to avoid solutions which have excessive size, compliance, and mass of long links. We could search a large family of mechanism designs to maximize M .

For example, if a design has five free parameters (typically link lengths and offsets), and we study ten possible values for each parameter, 10^5 designs must be evaluated.

Available computing power has grown much faster than the complexity of realizable mechanisms on the human scale (as measured by their degrees of freedom). Thus, brute-force search of design spaces is often sufficient, and sophisticated optimization techniques are not necessary.

Grounded Versus Ungrounded Devices

Most current devices that provide kinesthetic feedback are physically grounded, that is, forces felt by the operator are with respect to the operator's *ground*, such as the floor or desktop. Ungrounded haptic feedback devices are more mobile and can operate over larger workspaces compared to grounded devices, which enables them to be used in large-scale virtual environments. A number of ungrounded kinesthetic feedback devices have been developed, for example [30.31–33]. Comparisons have been made between the performance of ungrounded and grounded haptic displays [30.34]. Some ungrounded devices provide tactile rather than kinesthetic sensations, and these are described in Sect. 30.5.

30.2.2 Sensing

Haptic devices require sensors to measure the state of the device. This state may be modified by the operator's applied position/force, the haptic control law, and/or device and environment dynamics. The operator's input is sensed in the form of an applied position or an applied force. Sensing requirements for haptics are similar to those of other robotic devices (Chap. 20) so only haptics-specific sensing issues are discussed here.

Encoders

Rotary optical quadrature encoders are typically used as position sensors on the joints of haptic devices. They are often integrated with rotary motors, which serve as actuators. The underlying sensing mechanism for encoders is described in Sect. 20.1. The required resolution of an encoder for a haptic device depends on the ratio between the angular distance of a single encoder tick to the end-point motion in Cartesian space. The resolution of the selected position encoder has effects beyond simple spatial resolution of the endpoint, including the maximum stiffness that can be rendered (Sect. 30.4) without unstable or nonpassive behavior [30.35].

Many haptic applications, such as the rendering of virtual environments with damping (in which force is proportional to velocity), require velocity measurement. Velocity is typically obtained by numerical differentiation of the position signal obtained by an encoder. An algorithm for velocity estimation must be selected which is free of noise but minimizes phase lag at the frequencies of interest [30.36]. Thus, an alternative method is to use specialized hardware that measures the time between encoder ticks in order to compute the velocity [30.37].

Force Sensors

Force sensors are used in haptic devices as the operator input to an admittance-controlled device, or as a mechanism for canceling device friction and other undesirable dynamic properties in an impedance-controlled device. Force sensors are described in Sect. 20.4. When a force sensor such as a strain gauge or load cell measures the operator's applied force, care must be taken to thermally isolate the sensor, since thermal gradients in the sensor caused by body heat can affect force readings.

30.2.3 Actuation and Transmission

Haptic devices are differentiated from traditional computer input devices by actuators that are controlled to provide appropriate haptic sensations to the human operator. The performance of the haptic device depends heavily on the actuator properties and the mechanical transmission between the actuator and the haptic interaction point (HIP).

Requirements for Haptics

The primary requirements for actuators and mechanical transmission in impedance-type haptic devices are: low inertia, low friction, low torque ripple, back-driveability, and low backlash. In addition, if the design is such that the actuator itself moves as the user's position changes, a higher power-to-weight ratio is desired. Although closed-loop force control has been used for haptic display in impedance devices, most often the mechanism is designed to have sufficiently low friction and inertia so that open-loop force control is accurate enough.

One common mechanical transmission for haptic devices is the capstan drive (Fig. 30.6), which consists of smooth cables wrapped around pulleys of differing diameter to provide a gear ratio. A no-slip, high-friction contact between the cable and the pulleys is maintained through several wraps of the cable. The capstan drive minimizes friction forces felt by human operator be-

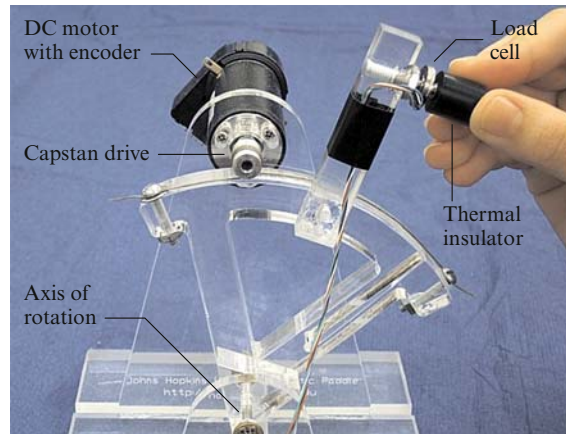


Fig. 30.6 This version of the Haptic Paddle [30.38] includes an encoder for position sensing, a single-axis load cell for force sensing, and a brushed motor with capstan transmission for actuation

cause it prevents translational forces on motor and joint axes.

Current amplifiers are typically used to create a direct relationship between the voltage output by the computer via a digital-to-analog (D/A) converter and the torque output by the motor. The effect of actuator and amplifier dynamics and D/A resolution on system stability is typically negligible in comparison to position sensor resolution and sampling rate for most haptic devices. Actuator or amplifier saturation can produce undesirable behavior, particularly in multi-degree-of-freedom haptic devices where a single saturated motor torque may change the apparent geometry of virtual objects. The force vector, and thus the corresponding actuator torques, must be scaled appropriately if any actuator is saturated.

30.2.4 An Example Device

As an illustrative example, we will provide detailed design information for a simple one-degree-of-freedom haptic device known as the Haptic Paddle [30.38]. This section is meant to provide a concrete description of the types of components that are used in kinesthetic haptic devices, and the device can also be constructed following the instructions provided by John Hopkins University [30.39]. Many widely available haptic devices share the common working principles of this device and differ chiefly in kinematic details arising from a greater number of degrees of freedom.

The haptic paddle shown in Fig. 30.6 is equipped with two sensors: a position encoder and a force sensor. A 500-counts-per-turn Hewlett-Packard HEDS 5540 encoder that is mounted directly on the motor. The quadrature process yields 2000 counts per revolution; the capstan transmission gear ratio and lever arm result in an position resolution of 2.24×10^{-5} m at the HIP. A PCI-Quad04 board from Measurement Computing is used to interface with the encoder. An Entran ± 10 N load cell (model ELFS-T3E-10N) is used to measure the applied operator force. A Delrin cap thermally insulates the load cell. In this device, the load cell can be used to minimize the effect of friction through a control law that attempts to zero the applied operator force when the HIP is not in contact with a virtual object. The signal from the load cell is passed through an instrumentation amplifier (Burr-Brown INA103) with a gain of 5 before it is read by the 16-bit A/D (PCI-DAS6402 configured for a range of ± 1.25 V).

The actuator is a brushed Maxon DC motor (model 118754m RE-25) with an aluminum pulley attached to the shaft. Like many commercial haptic devices, a cable is wrapped several times around the motor pulley and attached at each end of the large partial pulley. A PCI-DAS6402 data-acquisition card from Measurement Computing, is used to output voltages to the motor amplifier. The 16-bit D/A is configured for ± 10 V. The output of the D/A is passed through a current amplifier that gives a current through the motor that is proportional to the D/A voltage; the current amplifier is built around a National Semiconductor LM675 power operational amplifier. This gives direct control of applied torque on the motor. The resulting system has a force felt at the driving point that is proportional to the output voltage (1.65 N/V) during static operation. When the system is moving, the force applied to the operator may differ due to human and device dynamics.

30.3 Haptic Rendering

Haptic rendering (in impedance systems) is the process of computing the force required by contacts with virtual objects based on measurements of the operator's motion. This section describes haptic rendering for virtual environments. Haptic feedback for teleoperators is described in Chap. 31.

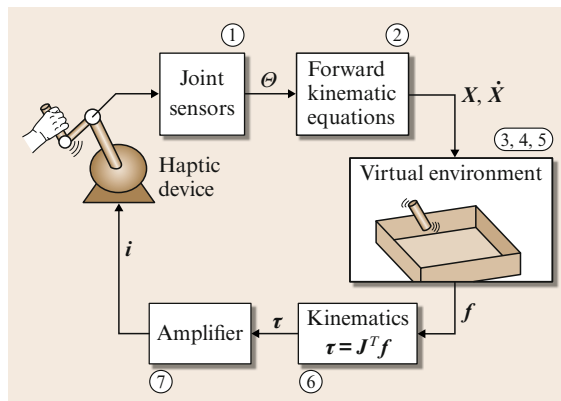


Fig. 30.7 Schematic diagram of the haptic rendering cycle for an impedance haptic display system. Virtual object moves in the virtual environment according to operator's displacement of the haptic device. Joint displacements (θ) sensed in the device (1) are processed through kinematics (2), collision detection (3), surface point determination (4), force calculation (5), kinematics (6), and actuation (7)

An important property of haptic systems is that their timing constraints are quite severe. To illustrate this point, tap a pencil on the table top. You hear a sound which is an audio representation of the contact dynamics between pencil tip and table top. Tactile receptors in the human fingers are known to respond up to 10 kHz [30.2]. To realistically render this type of contact between hard surfaces would require response well into the audio range of frequencies (up to 20 kHz) and thus sampling times of around 25 μ s. Even with specialized designs, haptic devices do not have this bandwidth, and such high fidelity is not usually a goal of haptic rendering. Achieving stability in any type of hard contact requires very high sampling rates. In practice, most haptic simulation systems are implemented with at least 1000 Hz sampling rate. This can be reduced to the low hundreds of Hertz if the virtual environment is limited to soft materials.

Basic Haptic Rendering

The computational process of haptic rendering can be formulated into the following seven sequential steps for each cycle (Fig. 30.7). The rendering cycle must typically be completed in under 1 ms for stability and realism:

1. Sensing (Sect. 30.2.2)
2. Kinematics
3. Collision detection

4. Determining surface point
5. Force calculation
6. Kinematics
7. Actuation (Sect. 30.2.3)

Kinematics

The position and velocity measurements acquired by sensors are typically in joint space. These must be converted through the forward kinematics model and the Jacobian matrix (Chap. 1) to the Cartesian position and velocity of the operator's hand or fingertip. In some applications the operator is virtually holding a tool or object whose shape is represented in the virtual environment but whose position and orientation are determined by the operator. In those others, the operator's fingertip or hand is represented by a point which makes only point contact with objects in the virtual environment (VE). We refer to a virtual handheld object as a virtual tool and, following [30.40], we refer to the single end-point as the haptic interaction point (HIP).

Collision Detection

For the point contact case, the collision detection software must determine if the position of the HIP at the current instant of time represents contact with a virtual object. In practice this usually means to determine if the HIP is penetrating or inside the object surface. The object surface is represented by a geometric model such as polygons or splines.

Although there is an extensive literature on collision detection in computer graphics, there are unique

aspects of the collision detection problem for haptics. In particular, speed of computation is paramount and worst-case speed, as opposed to average speed, is what counts. Solutions that evaluate in constant time are preferred. Section 30.3.1 addresses collision detection and haptic rendering for complex environments.

If the HIP is found to be outside all objects, then a force of zero is returned.

Determining Surface Point

Once it is determined that the HIP is inside a surface, the force to be displayed to the operator must be computed. Many researchers have used the idea of a virtual spring connecting the HIP to the nearest point on the surface as a model of interpenetration and force generation [30.40–42]. *Basdogan* and *Srinivasan* named this point the intermediate haptic interaction point (IHIP). However all of these authors realized that the closest surface point is not always the most faithful model of contact. For example, as the HIP moves laterally below the top surface of a cube (Fig. 30.8), eventually it becomes close enough to the edge that the closest surface point becomes the side of the cube. In this situation, the algorithm needs *memory* to keep the IHIP on the top surface and generate an upward force at all times or the operator is suddenly ejected out the side of the cube.

Force Calculation

Force is commonly computed by using the spring model (Hooke's Law)

$$f = kx, \quad (30.3)$$

where x is the vector from the HIP to the IHIP, and $k > 0$. When k is sufficiently large, the object surface will feel like a wall perpendicular to x . This *virtual wall*, or impedance surface, is a fundamental building block of most haptic virtual environments. Because the virtual wall is only displayed if a collision between the HIP and virtual object is detected, the wall is a unilateral constraint, governed by a nonlinear switching condition. As will be described in the following section, haptic virtual environments with complex geometries are often formed using a polygonal mesh in which each polygon is essentially a virtual wall. A virtual surface may also be allowed to deform globally, while the local interaction with the operator is governed by a virtual wall. *Virtual fixtures*, which are often constructed from virtual walls, can be overlaid on haptic feedback teleoperators to assist the operator during a teleoperated task (Chap. 31).

The pure stiffness model described above can be augmented to provide other effects, particularly through

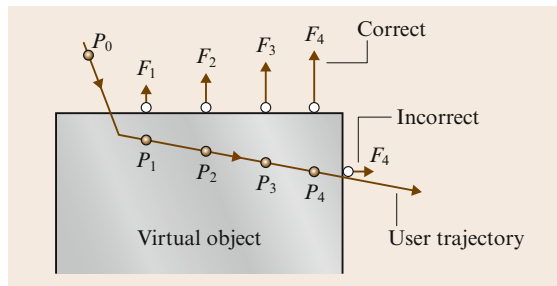


Fig. 30.8 Illustration of subtle aspects of haptic rendering of contact force. The operator fingertip trajectory enters object surface moving down and to right. Haptic interaction points (HIP) are shown at times 1–4 (solid circles, P_0 – P_4). Intermediate haptic interaction points (IHIP) are shown when the HIP is inside the object (open circles). At position P_4 , the algorithm must not render force based on the closest surface point or the operator will be ejected from the side of the object (feeling unnatural force tangential to the top surface)

the use of the *virtual coupling* described in Sect. 30.3.2. Damping can be added perpendicular or parallel to the surface. In addition, Coulomb or other nonlinear friction may be displayed parallel to the surface. To provide a more realistic display of hard surfaces, vibrations can also be displayed open loop at the moment of collision between the HIP and the surface, as is described in Sect. 30.5.

Kinematics

The computed force in Cartesian space must then be transformed into torques in the actuator space. Typically the calculation is

$$\boldsymbol{\tau} = \mathbf{J}^T \mathbf{f} \quad (30.4)$$

where $\boldsymbol{\tau}$ is the torque command to the actuators, \mathbf{f} is the desired force vector, and \mathbf{J}^T is the transpose of the haptic device Jacobian matrix (Chap. 1). If the haptic device has no dynamics and the actuators are perfect, the exact desired force is displayed to the operator. However, real device dynamics, time delays, and other nonidealities result in applied operator forces that differ from the desired forces.

30.3.1 Rendering Complex Environments

Today's computer power is sufficient that a variety of relatively simple algorithms can effectively render haptics for simple virtual environments, say consisting of a handful of simple geometric primitives such as spheres, cubes, and planes. However, the challenge is to scale these algorithms to complex environments such as those we are used to seeing in computer graphic renderings consisting of 10^5 – 10^7 polygons. A variety of approaches have been tried in the literature for efficient rendering of complex scenes. Zilles and Salisbury [30.41] found planar constraints due to the nearby surface polygons and solved for the closest IHIP point using Lagrange multipliers. Ruspini et al. [30.43] added force shading and friction models. Ho et al. [30.40] used a hierarchy of bounding spheres to determine the initial contact (collision) point, but thereafter searched neighboring surfaces, edges, and vertices (referred to as geometric primitives) of the current contacted triangle to find the closest point for the IHIP. Gregory et al. [30.44] imposed a discretization of three-dimensional (3-D) space onto the hierarchy to speed up the detection of initial contact with the surface. Johnson et al. [30.45] performed haptic rendering on moving models based on local extrema in distance between the model controlled by the haptic device and the rest of the scene. Lin and

Otaduy [30.46, 47] used level-of-detail representations of the objects for performing multiresolution collision detection, with the goal of satisfying real-time constraints while maximizing the accuracy of the computed proximity information.

Alternative algorithms and efficiencies can be derived by not representing the object surface as polygons. Thompson and Cohen [30.48] derived the mathematics for computing the interpenetration depth for surfaces directly from non-uniform rational B-spline (NURBS) models. McNeely et al. [30.18] took the extreme approach of voxelizing space at the millimeter scale. Each voxel contained a precomputed normal vector, stiffness properties, etc. 1000 Hz rendering was achieved with very complex CAD models containing millions of polygons (in the graphical equivalent representation), but large amounts of memory and precomputation are required. The performance of this algorithm is sufficiently high that it can be used to render hundreds of contact points simultaneously. This allows the operator to *hold* an arbitrarily shaped tool or object. The tool/object is populated by points surrounding its surface and the resultant force and moment on the tool is the sum of interaction forces computed on all of the surface points.

For surgical simulation, researchers have focused on the modeling and haptic rendering of the interaction between surgical instruments and organs. Researchers have attempted to model virtual tissue behavior in a wide variety of ways, which can be broadly classified as

1. linear elasticity based,
2. nonlinear (hyperelastic) elasticity-based finite-element (FE) methods
3. other techniques that are not based on FE methods or continuum mechanics

While most conventional linear and nonlinear FE algorithms cannot be run in real time, methods such as preprocessing can allow them to run at haptic rates [30.49]. Many researchers rely on data acquired from real tissues to model organ deformation and fracture accurately. Major challenges in this field include the modeling of connective tissue supporting the organ, friction between instruments and tissues, and topological changes occurring during invasive surgical procedures.

30.3.2 Virtual Coupling

So far we have rendered forces by computing the length and direction of a virtual spring and applying Hooke's Law (30.3). This spring is a special case of a *virtual coupling* [30.42] between the HIP and the IHIP. The vir-

tual coupling is an abstraction of the interpenetration model of force rendering. Instead of viewing the objects as compliant, we assume them to be rigid but connect them to the operator through a virtual spring. This imposes an effective maximum stiffness (that of the virtual coupling).

Problems with stable contact rendering (Sect. 30.4) often require more sophisticated virtual couplings than

a simple spring. For example, damping can be added, generalizing the force rendering model of (30.3) to

$$f = kx + b\dot{x}. \quad (30.5)$$

The parameters k and b can be empirically tuned for stable and high-performance operation. More-formal design methods for the virtual coupling are covered in Sect. 30.4.

30.4 Control and Stability of Haptic Interfaces

Introduction to the Problem

The haptic rendering system depicted in Fig. 30.7 is a closed-loop dynamical system. It is a challenge to render realistic contact forces yet retain the stable behavior of human-environment contact in the natural world. Instability in haptic interfaces manifests itself as buzzing, bouncing, or even wildly divergent behavior. The worst case for impedance devices is during attempted contact with stiff objects. Empirically, instability is frequently encountered when developing haptic interfaces with stiff virtual objects, but this instability can be eliminated by reducing the stiffness of the virtual object or by the operator making a firmer grasp on the haptic device.

Problem Description in Classical Control Terms

Although linear theory is of very limited use in applications, it can be applied to a basic analysis of the factors affecting instability [30.50]. Such a highly simplified model of an impedance device is shown in Fig. 30.9. $G_1(s)$ and $G_2(s)$ represent dynamics of the haptic device for both operator position sensing and force display respectively. Assume that the virtual environment and human operator/user (HO) can each be represented by a linear impedance such as

$$Z_{VE} = \frac{F_{VE}(s)}{X_{VE}(s)}, \quad (30.6)$$

$$Z_{HO} = \frac{F_{HO}(s)}{X_{HO}(s)}. \quad (30.7)$$

Then the loop gain of the closed-loop system from the human operator and back again is

$$G_I(s) = G_1(s)G_2(s)\frac{Z_{VE}(s)}{Z_{HO}(s)}. \quad (30.8)$$

Stability in the classical sense is assessed by applying the magnitude and phase criteria of Nyquist to $G_I(s)$. Increasing Z_{VE} (corresponding to stiffer or heavier virtual

objects) increases the magnitude of $G_I(s)$ and thus destabilizes the system while a firmer grasp by the human operator, which increases the magnitude of Z_{HO} , has a stabilizing effect. Similar arguments apply to phase shifts that might be present in any part of the system.

Limitations of Linear Theory

Although the model of Fig. 30.9 illustrates some qualitative features of haptic interface stability, linear continuous time theory is of little use in designing methods to stabilize the loop. Interesting virtual environments are nonlinear. Furthermore, they can rarely be linearized because applications often simulate discontinuous contact, for example, between a stylus in free space and a hard surface. A second feature is digital implementation, which introduces sampling and quantization—both of which have significant effects.

Sampling

Colgate et al. [30.42] incorporated consideration of discrete time sampling behavior in the stability analysis. They considered the problem of implementing a virtual wall of stiffness

$$H(z) = K + B\frac{z-1}{Tz}, \quad (30.9)$$

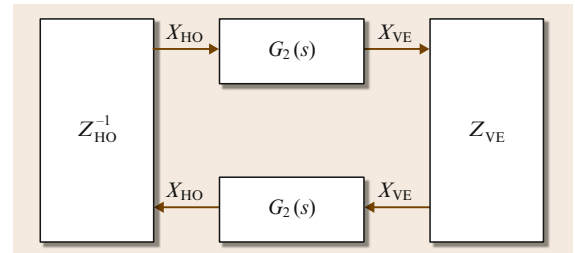


Fig. 30.9 Highly simplified linear model of haptic rendering to highlight some stability issues

where K is the virtual wall stiffness, B is the virtual wall damping coefficient, z is the z -transform variable, and T is the sampling time. They further modeled the haptic device (HD) in continuous time as

$$Z_{HD}(s) = \frac{1}{ms + b}, \quad (30.10)$$

where m and b are the mass and damping of the haptic device, respectively. They derived the following condition for passivity of the device

$$b > \frac{KT}{2} + |B|, \quad (30.11)$$

showing a significant stabilizing effect of high sampling rates and also of high mechanical damping in the haptic device.

Quantization

Additional factors include delays due to numerical integration schemes and quantization. These contributing factors to instability have been termed *energy leaks* by Gillespie and Cutkosky [30.51].

Passivity

Interesting virtual environments are always nonlinear and the dynamic properties of a human operator are important. These factors make it difficult to analyze haptic

systems in terms of known parameters and linear control theory. One fruitful approach is to use the idea of passivity to guarantee stable operation. Passivity is a sufficient condition for stability, and is reviewed more completely in Chap. 31 on telerobotics. There are many similarities between the control of haptic interfaces and bilateral teleoperation.

The major problem with using passivity for design of haptic interaction systems is that it is overly conservative, as shown in [30.35]. In many cases, performance can be poor if a fixed damping value is used to guarantee passivity under all operating conditions. Adams and Hannaford derived a method of virtual coupling design from two-port network theory which applied to all causality combinations and was less conservative than passivity based design [30.52]. They were able to derive optimal virtual coupling parameters using a dynamic model of the haptic device and by satisfying Lewellyn's *absolute stability criterion*, an inequality composed of terms in the two-port description of the combined haptic interface and virtual coupling system. Miller et al. derived another design procedure which extended the analysis to nonlinear environments and extracted a damping parameter to guarantee stable operation [30.53–55].

30.5 Tactile Display

Tactile displays are used to convey force, contact, and shape information to the skin. They purposely stimulate cutaneous receptors, with little effect on kinesthetic sensation. This is in contrast to the kinesthetic displays described earlier in this chapter, which focus on force display but also inherently provide cutaneous sensations through physical contact with a tool or thimble. Tactile displays are usually developed for a specific purpose, such as display of contact events, contact location, slip/shear, texture, and local shape. This is logical in some respects—as different types of cutaneous receptors, each with its own frequency response, receptive field, spatial distribution, and sensed parameter (e.g. local skin curvature, skin stretch, and vibration)—are associated with the different exploratory procedures described in Sect. 30.1.1.

In contrast to kinesthetic displays, the design and prototyping of tactile displays to render realistic contact information for virtual reality or teleoperation have proven very challenging. Accurate recreation of the local shape and pressure distribution at each fingertip requires a dense array of actuators. Devices specifi-

cally aimed at tactile perception are an active area of research, but most have not reached the stage of applications or commercial distribution, with the notable exception of Braille displays for the blind. In this section, we describe the various types of tactile displays, their design considerations, rendering algorithms, and applications.

30.5.1 Vibration Feedback

Vibration feedback is a popular method of providing tactile feedback. It can be used as a stand-alone method for haptic feedback or as an addition to a kinesthetic display. Vibrating elements, such as piezoelectric materials and small voice-coil motors, are lighter than the actuators used in kinesthetic devices, and can often be added to kinesthetic devices with little impact on existing mechanisms. In addition, high-bandwidth kinesthetic displays can be programmed to display open-loop vibrations through their normal actuators. The sensitivity for human vibration sensing ranges from DC to over 1 kHz, with peak sensitivity around 250 Hz.

We first consider the use of vibrations to convey impact or contact events—a technique that straddles kinesthetic and tactile feedback. When humans touch an environment, fast-acting sensors embedded in the skin record the minute vibrations occurring from this interaction. As described in Sect. 30.3, conventional approaches to haptic display usually consist of designing a virtual model with simple geometry, then using a first-order stiffness control law to emulate a surface. However, such first-order models often lack the realism of higher-order effects such as impact. With common haptic rendering algorithms, surfaces feel squishy or unrealistically smooth. One solution to improving the realism of such environments is to add higher-order effects such as textures and contact vibrations. These effects can use a library of surface models based on ad hoc analytical descriptions [30.56], which are sometimes tuned using qualitative operator feedback, physical measurements (reality-based models created from empirical data) [30.57, 58], or a combination of the two [30.59]. At the instant collision is detected between the HIP and a surface of the virtual object, the appropriate waveform is called out of the library, scaled according to the context of the motion (such as velocity or acceleration), and played open loop through an actuator. That actuator may be the same one simultaneously displaying lower-frequency force information, as shown in

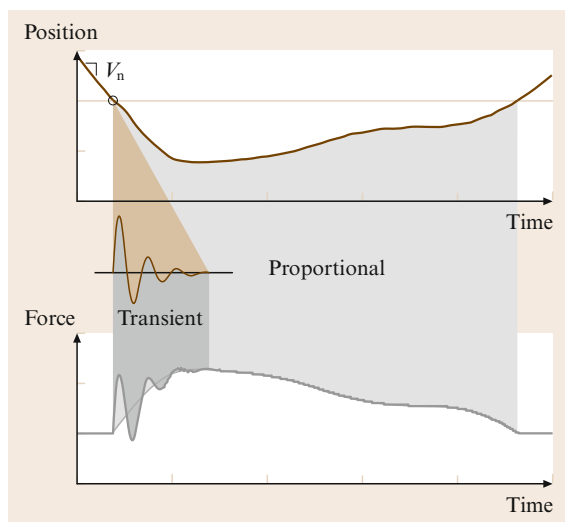


Fig. 30.10 Event-triggered open-loop force signals superimpose on traditional penetration-based feedback forces, providing vibration feedback that improves the realism of hard surfaces in virtual environments (after [30.60] with permission)

Fig. 30.10, or it might be a separate transducer. *Kuchenbecker et al.* [30.60] considered the dynamics of the haptic device to display the most accurate vibration waveforms possible, and compared a number of different vibration waveform generation techniques meant to convey impact superimposed on force feedback in virtual environments. Most of the vibration feedback methods performed similarly in terms of realism, and they were also significantly more realistic than conventional force feedback alone.

Vibration feedback can also be used to provide information about patterned textures, roughness, and other phenomena that have clear vibratory signals. This type of vibration feedback is often termed *vibrotactile* feedback. In teleoperated environments, *Kontarinis and Howe* [30.61] showed that damaged ball bearings could be identified through vibration feedback, and *Dennerlein et al.* [30.62] demonstrated that vibration feedback improved performance over no haptic feedback, for a telemanipulation task inspired by undersea field robotics. In these teleoperator systems, vibration-sensitive sensors such as accelerometers and piezoelectric sensors are used to pick up and, in most cases, directly provide the vibration signals as inputs to a vibrotactile actuator. In virtual environments, *Okamura et al.* [30.63] displayed vibrations modeled based on textures and puncture of a membrane. Similar to the event-based haptics above, the vibration waveforms were modeled based on earlier experiments and played open loop during interaction with the virtual environment.

Finally, vibration feedback has been used as a method of sensory substitution, to convey direction, attention, or other information, e.g., [30.64–66]. In this case, the strength and clarity of the signal, not realism, is the goal. Vibration frequencies near the peak human sensitivity are most effective, and there exist commercially available vibrotactile actuators (or *tactors*) suitable for such applications, e.g., Engineering Acoustic, Inc.’s C2 Tactor [30.67]. Elements within an array of tactors can be selectively turned on and off to evoke the sensory saltation phenomenon, in which a pattern of brief pulses on a series of tactors is perceived not as successive taps at different locations, but as a single tap that is traveling or hopping over the skin.

30.5.2 Contact Location, Slip, and Shear Display

In early work related to robotic dexterous manipulation, it was found that knowledge of the contact point

between a robot hand and a grasped object is essential for manipulation. Without this knowledge, the robot will easily drop the object due to rapid accumulation of grasp errors. While many robotics researchers and some companies have developed tactile array sensors capable of measuring contact location, pressure distribution, and local object geometry (Chap. 19), practical methods for display of this information to the human operator of a virtual or teleoperated environment have proven much more difficult. We begin our discussion of contact display by considering contact location, slip, and shear display, which have the common goal of displaying the motion of a single area of contact relative to the skin (almost invariably on a finger). Arrays of pins that rise and fall to create a pressure distribution on the skin have been the most popular method to date for displaying contact information, but we will address those designs in the following section on local shape, since their primary advantage is the display of spatially distributed information. Instead, we will focus here on tactile devices that are designed to specifically address the problem of contact location and motion.

As an example of contact location display, *Provancher et al.* [30.68] developed a system that renders the location of the contact centroid moving on the user's fingertip. The tactile element is a free-rolling cylinder that is normally suspended away from the fingertip, but comes into contact with the skin when the operator pushes on a virtual object. The motion of the cylinder over the skin is controlled by sheathed push-pull wires. This allows the actuators to be placed remotely, creating a lightweight, thimble-sized package that can be unobtrusively mounted on a kinesthetic haptic device. An experiment demonstrated that human operators performed similarly during real manipulation and virtual manipulation (using the tactile display) in an object curvature discrimination task. In addition, operators were able to use the device to distinguish between manipulations of rolling and anchored but rotating virtual objects.

Humans use slip and incipient slip widely during manipulation tasks [30.69]. To reproduce these sensations for experiments to characterize human slip sensation, researchers have created stand-alone one-degree-of-freedom slip displays [30.70–72]. *Webster et al.* [30.72] created a two-degree-of-freedom tactile slip display, which uses an actuated rotating ball positioned under the user's fingertip. The lightweight, modular tactile display can be attached to a multi-degree-of-freedom kinesthetic interface and used to display virtual environments with slip. Experimental results demonstrate that operators

complete a virtual manipulation task with lower applied forces using combined slip and force feedback in comparison with conventional force feedback alone. Skin stretch can also be integrated with a slip display to provide information about pre-slip conditions. For example, *Tsagarakis et al.* [30.73] developed a lightweight device that uses a V configuration of miniature motors to provide sensations of relative lateral motion (direction and velocity) onto the operator's fingertips. Generation of two-dimensional slip/stretch is achieved by coordinating the rotational speed and direction of the two motors.

A novel device that uses slip and friction to display compelling tactile sensations is the TPd (tactile pattern display) [30.74]. Ultrasonic frequency, low amplitude vibrations of a flat plate create a film of air between the plate and a human finger touching the plate, thereby reducing friction. The 33 kHz vibration of the plate cannot be perceived by the human. The amount of friction reduction varies with vibration amplitude, allowing indirect control of shear forces on the finger during active exploration. Finger position and velocity feedback enables haptic rendering of spatial texture sensations.

30.5.3 Local Shape

Most tactile devices for local shape display consist of an array of individual pin elements that move normal to the surface. Often, a layer of elastic material is used to cover the pins so that the operator contacts a smooth surface rather than the pins directly. Other systems use individual elements that move laterally, and some substitute electrodes for moving parts to form an array of electrocutaneous elements. A number of researchers have used psychophysical and perceptual experimental results to define design parameters such as number of pins, spacing, and amplitude of pin-based tactile displays. A commonly used metric is the two-point discrimination test, which defines the minimum distance between two contact points on the skin at which they are perceived as two, rather than one point. This discrimination limit varies widely for skin on different parts of the body, with the fingertips having one of the smallest (usually cited as less than 1 mm, although this depends on the shape and size of the contacts) [30.75, 76]. *Moy et al.* [30.77] quantified several perceptual capabilities of the human tactile system based on predicted subsurface strain and psychophysical experiments that measured amplitude resolution, the effects of shear stress, and the effects of viscoelasticity (creep and relaxation) on tactile perception for static touch. They found that 10%

amplitude resolution is sufficient for a teletaction system with a 2 mm elastic layer and 2 mm tacter spacing. A different type of experiment examines the kind of tactile information relevant to a particular application. For example, *Peine and Howe* [30.78] found that sensed deformation of the fingerpad, and not changes in pressure distribution, were responsible for localizing lumps in a soft material, such as a tumor in tissue.

We will now highlight a few distinctive designs of array-type tactile displays. A number of actuator technologies have been applied to create tactile arrays, including piezoelectric, shape-memory alloy (SMA), electromagnetic, pneumatic, electrorheological, micro-electromechanical system (MEMS), and electrotactile. Further reading on tactile display design and actuation is available in review papers, including [30.80–85].

We will first consider two ends of the spectrum in complexity/cost of the pin-based approach. *Killebrew et al.* [30.86] developed a 400-pin, 1 cm² tactile stimulator to present arbitrary spatiotemporal stimuli to the skin for neuroscience experiments. Each pin is under independent computer control and can present over 1200 stimuli per minute. While not practical for most haptic

applications due to the size and weight of the actuation unit, it is the highest-resolution tactile display built to date and can be used to evaluate potential designs for lower-resolution displays. *Wagner et al.* [30.79] created a 36-pin, 1 cm² tactile shape display that uses commercially available radio-controlled (RC) servomotors. The display can represent maximum frequencies of 7.5–25 Hz, pending on the amount of pin deflection, and is shown in Fig. 30.11. *Howe et al.* [30.87, 88] have also explored the use of shape-memory alloys for pin actuation, for the application of remote palpation.

In contrast to pins that move normal to the surface, recent tactile array designs have incorporated pins that move laterally. First introduced by *Hayward and Cruz-Hernandez* [30.89], the most recent compact, lightweight, modular design [30.90] uses a 6 × 10 piezo bimorph actuator array with a spatial resolution of 1.8 mm × 1.2 mm. The force of the individual actuators provides sufficient skin pad motion/stretch to excite mechanoreceptors [30.91]. A pilot test demonstrated that subjects could detect a virtual line randomly located on an otherwise smooth virtual surface, and the device has also been tested as a Braille display [30.92]. Another lateral stretch display and its evaluation is described in [30.93]. Other novel approaches to tactile display include sending small currents through the skin or tongue using an electrocutaneous array [30.94] and the application of air pressure to stimulate only superficial mechanoreceptors [30.95].

30.5.4 Temperature

Because the human body is typically warmer than objects in the environment, thermal perceptions are based on a combination of thermal conductivity, thermal capacity, and temperature. This allows us to infer not only temperature difference, but also material composition [30.96]. Most thermal display devices are based on thermoelectric coolers, also known as Peltier heat pumps. Thermoelectric coolers consist of a series of semiconductor junctions connected electrically in series and thermally in parallel. The thermoelectric cooler is designed to pump heat from one ceramic faceplate to the other, but if used in reverse, a temperature gradient across the device produces a proportional potential; as a measure of relative temperature change. The designs of haptic thermal displays mostly use off-the-shelf components, and their applications are typically straightforward, enabling identification of objects in a virtual or teleoperated environment by their temperature and thermal conductivity.

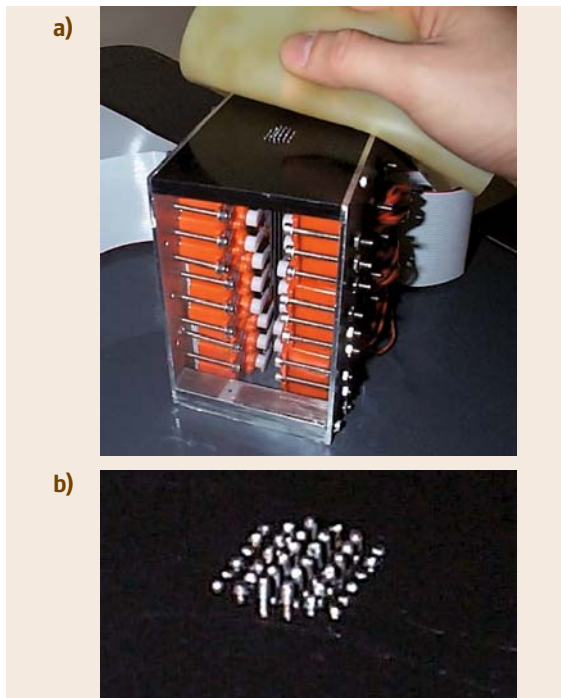


Fig. 30.11 (a) A low-cost 36-pin tactile display using RC servomotor actuation. (b) A closeup of the 6 × 6 display, showing a sine wave grating (after [30.79] with permission)

Ho and Jones [30.97] provide a review of haptic temperature display, as well as promising results suggesting that a thermal display is capable of facilitating object recognition when visual cues are limited. Although numerous systems have integrated thermal display with other types of haptic display, the Data Glove Input System designed by *Caldwell et al.* [30.98, 99] was one of the first to do so. Their haptic interface provides simultaneous force, tactile, and thermal feedback. The Peltier device used for thermal display contacts the dorsal surface of the index finger. Subjects achieved a 90%

success rate in identifying materials such as a cube of ice, a soldering iron, insulating foam, and a block of aluminum, based only on thermal cues. The study of human temperature perception is particularly interesting, including issues such as spatial summation and the psychological relevance of temperature display. For example, in prosthetic limbs, temperature display may be useful not only for practical reasons such as safety and material identification, but also for reasons of personal comfort, such as feeling the warmth of a loved one's hand.

30.6 Conclusions and Further Reading

Haptic technology, which attempts to provide compelling sensations to human operators in virtual and teleoperated environments, is a relatively new, but fast-growing and dynamic area of research. The field relies not only on fundamental foundations from robotics and control theory, but also on fields in the human sciences, particularly neuroscience and psychology. To date, commercial success of haptics has been in the areas of entertainment, medical simulation, and design, although novel devices and applications are regularly appearing.

There exist many books on the topic of haptic technology, most of them compendiums from workshops or conferences on the subject. One of the earliest books on haptics, by *Burdea* [30.100], provides a thorough review of applications and haptic devices up to 1996. A book specifically focused on haptic rendering, designed for potential use as a textbook, has been edited by *Ming Lin* and *Miguel Otaduy* [30.101]. In addition, we recommend the following useful articles: *Hayward* and *MacLean* [30.102, 103] describe the fundamentals of constructing experimental haptic devices of modest

complexity, the software components needed to drive them, and the interaction design concepts important to creating usable systems. *Hayward, et al.* [30.104] also provide a tutorial on haptic devices and interfaces. *Salisbury, et al.* [30.105] describe the basic principles of haptic rendering. *Hayward* [30.106] describes a number of tactile illusions that can inspire creative solutions for haptic interface design. *Robles-De-La-Torre* [30.107] underscores the importance of haptics with compelling examples of humans who have lost the sense of touch.

Finally, there exist two journals that are specific to the field of haptics: *Haptics-e* (www.haptics-e.org) and *IEEE Transactions on Haptics* (first issue expected 2008). Several conferences are specifically devoted to haptics: Eurohaptics and the Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems are held separately in even years, and the on odd years become a single conference, World Haptics. The *IEEE* technical committee on haptics (www.worldhaptics.org) provides information about relevant publication forums.

References

- 30.1 G.A. Gescheider: *Psychophysics: The Fundamentals* (Lawrence Erlbaum, New York 1985)
- 30.2 K.B. Shimoga: A survey of perceptual feedback issues in dexterous telemanipulation. I. Finger force feedback, *Proc. Virtual Real. Annu. Int. Symp.* (1993) pp. 263–270
- 30.3 M.A. Srinivasan, R.H. LaMotte: Tactile discrimination of shape: responses of slowly and rapidly adapting mechanoreceptive afferents to a step indented into the monkey fingerpad, *J. Neurosci.* 7(6), 1682–1697 (1987)
- 30.4 R.H. LaMotte, R.F. Friedman, C. Lu, P.S. Khalsa, M.A. Srinivasan: Raised object on a planar surface stroked across the fingerpad: Responses of cutaneous mechanoreceptors to shape and orientation, *J. Neurophysiol.* 80, 2446–2466 (1998)
- 30.5 R.H. LaMotte, J. Whitehouse: Tactile detection of a dot on a smooth surface: peripheral neural events, *J. Neurophysiol.* 56, 1109–1128 (1986)
- 30.6 R. Hayashi, A. Miyake, H. Jijiwa, S. Watanabe: Postural readjustment to body sway induced by vibration in man, *Exp. Brain Res.* 43, 217–225 (1981)

- 30.7 G.M. Goodwin, D.I. McCloskey, P.B.C. Matthews: The contribution of muscle afferents to kinesthesia shown by vibration induced illusions of movement an the effects of paralysing joint afferents, *Brain* **95**, 705–748 (1972)
- 30.8 G.S. Dhillon, K.W. Horch: Direct neural sensory feedback and control of a prosthetic arm, *IEEE Trans. Neural Syst. Rehabil. Eng.* **13**(4), 468–472 (2005)
- 30.9 L.A. Jones: Perception and control of finger forces, *DSC* (1998) pp.133–137
- 30.10 S. Lederman, R. Klatzky: Hand movements: a window into haptic object recognition, *Cognit. Psychol.* **19**(3), 342–368 (1987)
- 30.11 R. Klatzky, S. Lederman, V. Metzger: Identifying objects by touch, An ‘expert system’, *Percept. Psychophys.* **37**(4), 299–302 (1985)
- 30.12 S. Lederman, R. Klatzky: Haptic classification of common objects: Knowledge-driven exploration, *Cognit. Psychol.* **22**, 421–459 (1990)
- 30.13 O.S. Bholat, R.S. Haluck, W.B. Murray, P.G. Gorman, T.M. Krummel: Tactile feedback is present during minimally invasive surgery, *J. Am. Coll. Surg.* **189**(4), 349–355 (1999)
- 30.14 C. Basdogan, S. De, J. Kim, M. Muniyandi, M.A. Srinivasan: Haptics in minimally invasive surgical simulation and training, *IEEE Comput. Graphics Appl.* **24**(2), 56–64 (2004)
- 30.15 P. Strom, L. Hedman, L. Sarna, A. Kjellin, T. Wredmark, L. Fellander-Tsai: Early exposure to haptic feedback enhances performance in surgical simulator training: a prospective randomized crossover study in surgical residents, *Surg. Endosc.* **20**(9), 1383–1388 (2006)
- 30.16 A. Liu, F. Tendick, K. Cleary, C. Kaufmann: A survey of surgical simulation: applications, technology, and education, *Presence Teleop. Virt. Environ.* **12**(6), 599–614 (2003)
- 30.17 R.M. Satava: Accomplishments and challenges of surgical simulation, *Surg. Endosc.* **15**(3), 232–241 (2001)
- 30.18 W.A. McNeely, K.D. Puterbaugh, J.J. Troy: Six degree-of-freedom haptic rendering using voxel sampling, *Proc. SIGGRAPH 99* (1999) pp. 401–408
- 30.19 SensAble Technologies: www.sensable.com (Woburn 2007)
- 30.20 Immersion Corporation: www.immersion.com (San Jose 2007)
- 30.21 T.H. Massie, J.K. Salisbury: The phantom haptic interface: A device for probing virtual objects, *Proc. ASME Dyn. Syst. Contr. Div.*, Vol. 55 (1994) pp. 295–299
- 30.22 Novint Technologies: www.novint.com (Albuquerque 2007)
- 30.23 M.C. Cavusoglu, D. Feygin, F. Tendick: A critical study of the mechanical and electrical properties of the PHANTOM haptic interface and improvements for high-performance control, *Presence* **11**(6), 555–568 (2002)
- 30.24 R.Q. van der Linde, P. Lammerste, E. Frederiksen, B. Ruiters: The HapticMaster, a new high-performance haptic interface, *Proc. Eurohaptics Conf.* (2002) pp.1–5
- 30.25 T. Yoshikawa: Manipulability of robotic mechanisms, *Int. J. Robot. Res.* **4**(2), 3–9 (1985)
- 30.26 J.K. Salisbury, J.T. Craig: Articulated hands: Force control and kinematics issues, *Int. J. Robot. Res.* **1**(1), 4–17 (1982)
- 30.27 P. Buttolo, B. Hannaford: Pen based force display for precision manipulation of virtual environments, *Proc. VRAIS-95* (1995) pp. 217–225
- 30.28 P. Buttolo, B. Hannaford: Advantages of actuation redundancy for the design of haptic displays, *Proc. ASME Fourth Annu. Symp. Haptic Interf. Virt. Environ. Teleop. Syst.*, Vol. DSC-57-2 (1995) pp. 623–630
- 30.29 T. Yoshikawa: *Foundations of Robotics* (MIT Press, Cambridge 1990)
- 30.30 S. Venema, B. Hannaford: A probabilistic representation of human workspace for use in the design of human interface mechanisms, *IEEE Trans. Mechatron.* **6**(3), 286–294 (2001)
- 30.31 H. Yano, M. Yoshie, H. Iwata: development of a non-grounded haptic interface using the gyro effect, *Proc. 11th Symp. Haptic Interf. Virt. Environ. Teleop. Syst.* (2003) pp. 32–39
- 30.32 C. Swindells, A. Unden, T. Sang: TorqueBAR: an ungrounded haptic feedback device, *Proc. 5th Int. Conf. Multimodal Interf.* (2003) pp. 52–59
- 30.33 Immersion Corporation: CyberGrasp – Ground-breaking haptic interface for the entire hand (last accessed 2006) www.immersion.com/3d/products/cyber_grasp.php
- 30.34 C. Richard, M.R. Cutkosky: Contact force perception with an ungrounded haptic interface, 1997 ASME IMECE 6th Annu. Symp. Haptic Interf. (1997)
- 30.35 J.J. Abbott, A.M. Okamura: Effects of position quantization and sampling rate on virtual-wall passivity, *TRO* **21**(5), 952–964 (2005)
- 30.36 S. Usui, I. Amidror: Digital low-pass differentiation for biological signal processing, *IEEE Trans. Biomedic. Eng.* **BME-29**(10), 686–693 (1982)
- 30.37 P. Bhatti, B. Hannaford: Single chip optical encoder based velocity measurement system, *IEEE Trans. Contr. Syst. Technol.* **5**(6), 654–661 (1997)
- 30.38 A.M. Okamura, C. Richard, M.R. Cutkosky: Feeling is believing: Using a force-feedback joystick to teach dynamic systems, *ASME J. Eng. Educ.* **92**(3), 345–349 (2002)
- 30.39 John Hopkins University: <http://haptics.jhu.edu/paddle> (Baltimore)
- 30.40 C.H. Ho, C. Basdogan, M.A. Srinivasan: Efficient point-based rendering techniques for haptic display of virtual objects, *Presence* **8**, 477–491 (1999)

- 30.41 C.B. Zilles, J.K. Salisbury: A constraint-based god-object method for haptic display, *IROS* (1995) pp. 146–151
- 30.42 J.E. Colgate, M.C. Stanley, J.M. Brown: Issues in the haptic display of tool use, *IROS* (1995) pp. 140–145
- 30.43 D. Ruspini, O. Khatib: Haptic display for human interaction with virtual dynamic environments, *J. Robot. Syst.* **18**(12), 769–783 (2001)
- 30.44 A. Gregory, A. Mascarenhas, S. Ehmann, M. Lin, D. Manocha: Six degree-of-freedom haptic display of polygonal models, *Proc. Vis.* 2000 (2000) pp. 139–146
- 30.45 D.E. Johnson, P. Willemsen, E. Cohen: 6-DOF haptic rendering using spatialized normal cone search, *Trans. Vis. Comput. Graphics* **11**(6), 661–670 (2005)
- 30.46 M.A. Otaduy, M.C. Lin: A modular haptic rendering algorithm for stable and transparent 6-DOF manipulation, *IEEE Trans. Vis. Comput. Graphics* **22**(4), 751–762 (2006)
- 30.47 M.C. Lin, M.A. Otaduy: Sensation-preserving haptic rendering, *IEEE Comput. Graphics Appl.* **25**(4), 8–11 (2005)
- 30.48 T. Thompson, E. Cohen: Direct haptic rendering of complex trimmed NURBS models, 8th Annu. Symp. Haptic Interf. Virt. Environ. Teleop. Syst. (1999)
- 30.49 S.P. DiMaio, S.E. Salcudean: Needle insertion modeling and simulation, *IEEE Trans. Robot. Autom.* **19**(5), 864–875 (2003)
- 30.50 B. Hannaford: Stability and performance tradeoffs in bi-lateral telemanipulation, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 3 (1989) pp. 1764–1767
- 30.51 B. Gillespie, M. Cutkosky: Stable user-specific rendering of the virtual wall, *Proc. ASME Int. Mech. Eng. Conf. Expo.*, Vol. DSC-58 (1996) pp. 397–406
- 30.52 R.J. Adams, B. Hannaford: Stable haptic interaction with virtual environments, *IEEE Trans. Robot. Autom.* **15**(3), 465–474 (1999)
- 30.53 B.E. Miller, J.E. Colgate, R.A. Freeman: Passive implementation for a class of static nonlinear environments in haptic display, *Proc. IEEE Int. Conf. Robot. Automation* (1999) pp. 2937–2942
- 30.54 B.E. Miller, J.E. Colgate, R.A. Freeman: Computational delay and free mode environment design for haptic display, *Proc. ASME Dyn. Syst. Cont. Div.* (1999)
- 30.55 B.E. Miller, J.E. Colgate, R.A. Freeman: Environment delay in haptic systems, *Proc. IEEE Int. Conf. Robot. Autom.* (2000) pp. 2434–2439
- 30.56 S.E. Salcudean, T.D. Vlaar: On the emulation of stiff walls and static friction with a magnetically levitated input/output device, *ICRA*, Vol. 119 (1997) pp. 127–132
- 30.57 P. Wellman, R.D. Howe: Towards realistic vibrotactile display in virtual environments, *Proc. 4th Symp. Haptic Interf. Virt. Environ. Teleop. Syst. ASME Int. Mech. Eng. Congress Expo.* (1995) pp. 713–718
- 30.58 K. MacLean: The haptic camera: A technique for characterizing and playing back haptic properties of real environments, *Proc. 5th Annu. Symp. Haptic Interf. Virt. Environ. Teleop. Syst. ASME/IMECE* (1996)
- 30.59 A.M. Okamura, J.T. Dennerlein, M.R. Cutkosky: Reality-based models for vibration feedback in virtual environments, *ASME/IEEE Trans. Mechatron.* **6**(3), 245–252 (2001)
- 30.60 K.J. Kuchenbecker, J. Fiene, G. Niemeyer: Improving contact realism through event-based haptic feedback, *IEEE Trans. Vis. Comput. Graphics* **12**(2), 219–230 (2006)
- 30.61 D.A. Kontarinis, R.D. Howe: Tactile display of vibratory information in teleoperation and virtual environments, *Presence* **4**(4), 387–402 (1995)
- 30.62 J.T. Dennerlein, P.A. Millman, R.D. Howe: Vibrotactile feedback for industrial telemanipulators, *Proc. ASME Dyn. Syst. Contr. Div.*, Vol. 61 (1997) pp. 189–195
- 30.63 A.M. Okamura, J.T. Dennerlein, R.D. Howe: Vibration feedback models for virtual environments, *Proc. IEEE Int. Conf. Robot. Autom.* (1998) pp. 674–679
- 30.64 R.W. Lindeman, Y. Yanagida, H. Noma, K. Hosaka: Wearable vibrotactile systems for virtual contact and information display, *Virt. Real.* **9**(2–3), 203–213 (2006)
- 30.65 C. Ho, H.Z. Tan, C. Spence: Using spatial vibrotactile cues to direct visual attention in driving scenes, *Transp. Res. F Traffic Psychol. Behav.* **8**, 397–412 (2005)
- 30.66 H.Z. Tan, R. Gray, J.J. Young, R. Traylor: A haptic back display for attentional and directional cueing, *Haptics-e Electron. J. Haptics Res.* **3**(1), 20 (2003)
- 30.67 C2 Tactor: Engineering Acoustic Inc.: www.eaiinfo.com (Casselberry 2007)
- 30.68 W.R. Provancher, M.R. Cutkosky, K.J. Kuchenbecker, G. Niemeyer: Contact location display for haptic perception of curvature and object motion, *Int. J. Robot. Res.* **24**(9), 691–702 (2005)
- 30.69 R.S. Johansson: Sensory input and control of grip, *Novartis Foundat. Symp.*, Vol. 218 (1998) pp. 45–59
- 30.70 K.O. Johnson, J.R. Phillips: A rotating drum stimulator for scanned embossed patterns and textures across the skin, *J. Neurosci. Methods* **22**, 221–231 (1998)
- 30.71 M.A. Salada, J.E. Colgate, P.M. Vishton, E. Frankel: Two experiments on the perception of slip at the fingertip, 12th Symp. Haptic Interf. Virt. Environ. Teleop. Syst. (2004) pp. 472–476
- 30.72 R.J. Webster III, T.E. Murphy, L.N. Verner, A.M. Okamura: A novel two-dimensional tactile slip display: Design, kinematics and perceptual experiment, *ACM Trans. Appl. Percept.* **2**(2), 150–165 (2005)
- 30.73 N.G. Tsagarakis, T. Horne, D.G. Caldwell: SLIP AESTHEASIS: a portable 2D slip/skin stretch display for the fingertip, *First Joint Eurohaptics Conf. Symp.*

- Haptic Interf. Virt. Environ. Teleop. Syst. (World Haptics) (2005) pp. 214–219
- 30.74 L. Winfield, J. Glassmire, J. E. Colgate, M. Peshkin: T-PaD: Tactile Pattern Display through Variable Friction Reduction. Second Joint Eurohaptics Conf. Symp. Haptic Interf. Virt. Environ. Teleop. Syst. (World Haptics) (2007) pp. 421–426
- 30.75 K.O. Johnson, J.R. Phillips: Tactile spatial resolution. I. Two-point discrimination, gap detection, grating resolution, and letter recognition, *J. Neurophysiol.* **46**(6), 1177–1192 (1981)
- 30.76 N. Asamura, T. Shinohara, Y. Tojo, N. Koshida, H. Shinoda: Necessary spatial resolution for realistic tactile feeling display, *IEEE Int. Conf. Robot. Autom.* (2001) pp. 1851–1856
- 30.77 G. Moy, U. Singh, E. Tan, R.S. Fearing: Human psychophysics for teletaction system design, *Haptics-e Electron. J. Haptics Res.* **1**(3) (2000)
- 30.78 W.J. Peine, R.D. Howe: Do humans sense finger deformation or distributed pressure to detect lumps in soft tissue, *Proc. ASME Dyn. Syst. Contr. Div. ASME Int. Mech. Eng. Congress Expo.*, Vol. DSC-64 (1998) pp. 273–278
- 30.79 C.R. Wagner, S.J. Lederman, R.D. Howe: Design and performance of a tactile shape display using RC servomotors, *Haptics-e Electron. J. Haptics Res.* **3**(4) (2004)
- 30.80 K.B. Shimoga: A survey of perceptual feedback issues in dexterous telemanipulation: Part II, Finger touch feedback, *Proc. IEEE Virt. Real. Annu. Int. Symp.* (1993) pp. 271–279
- 30.81 K.A. Kaczmarek, P. Bach-Y-Rita: Tactile displays. In: *Virtual Environments and Advanced Interface Design*, ed. by W. Barfield, T.A. Furness (Oxford Univ. Press, Oxford 1995) pp. 349–414
- 30.82 M. Shimojo: Tactile sensing and display, *Trans. Inst. Electr. Eng. Jpn. E* **122**, 465–468 (2002)
- 30.83 S. Tachi: Roles of tactile display in virtual reality, *Trans. Inst. Electr. Eng. Jpn. E* **122**, 461–464 (2002)
- 30.84 P. Kammermeier, G. Schmidt: Application-specific evaluation of tactile array displays for the human fingertip, *IEEE/RSJ Int. Conf. Intell. Robot. Syst. Int. Conf. Intell. Robot. Syst.* (2002)
- 30.85 S.A. Wall, S. Brewster: Sensory substitution using tactile pin arrays: human factors, technology and applications, *Signal Process.* **86**(12), 3674–3695 (2006)
- 30.86 J.H. Killebrew, S.J. Bensmaia, J.F. Dammann, P. Denchev, S.S. Hsiao, J.C. Craig, K.O. Johnson: A dense array stimulator to generate arbitrary spatio-temporal tactile stimuli, *J. Neurosci. Methods* **161**(1), 62–74 (2007)
- 30.87 R.D. Howe, W.J. Peine, D.A. Kontarinis, J.S. Son: Remote palpation technology, *IEEE Eng. Med. Biol.* **14**(3), 318–323 (1995)
- 30.88 P.S. Wellman, W.J. Peine, G. Favalora, R.D. Howe: Mechanical design and control of a high-bandwidth shape memory alloy tactile display, *Exp. Robot. V* **232**, 56–66 (1998)
- 30.89 V. Hayward, M. Cruz-Hernandez: Tactile display device using distributed lateral skin stretch, *Symp. Haptic Interf. Virt. Environ. Teleop. Syst. (ASME IMECE)*, Vol. DSC-69-2 (2000) pp. 1309–1314
- 30.90 Q. Wang, V. Hayward: Compact, portable, modular, high-performance, distributed tactile transducer device based on lateral skin deformation, *14th Symp. Haptic Interf. Virt. Environ. Teleop. Syst.* (2006) pp. 67–72
- 30.91 Q. Wang, V. Hayward: In vivo biomechanics of the fingerpad skin under local tangential traction, *J. Biomech.* **40**(4), 851–860 (2007)
- 30.92 V. Levesque, J. Pasquero, V. Hayward: Braille display by lateral skin deformation with the STReSS2 tactile transducer, *Second Joint Eurohaptics Conf. Symp. Haptic Interf. Virt. Environ. Teleop. Syst. (World Haptics)* (2007) pp. 115–120
- 30.93 K. Drewing, M. Fritschi, R. Zopf, M.O. Ernst, M. Buss: First evaluation of a novel tactile display exerting shear force via lateral displacement, *ACM Trans. Appl. Percept.* **2**(2), 118–131 (2005)
- 30.94 K.A. Kaczmarek, J.G. Webster, P. Bach-Y-Rita, W.J. Tompkins: Electrotactile and vibrotactile displays for sensory substitution systems, *IEEE Trans. Biomed. Eng.* **38**, 1–16 (1991)
- 30.95 N. Asamura, N. Yokoyama, H. Shinoda: Selectively stimulating skin receptors for tactile display, *IEEE Comput. Graphics Appl.* **18**, 32–37 (1998)
- 30.96 H.-N. Ho, L.A. Jones: Contribution of thermal cues to material discrimination and localization, *Percept. Psychophys.* **68**, 118–128 (2006)
- 30.97 H.-N. Ho, L.A. Jones: Development and evaluation of a thermal display for material identification and discrimination, *ACM Trans. Appl. Percept.* **4**(2), 118–128 (2007)
- 30.98 D.G. Caldwell, C. Gosney: Enhanced tactile feedback (tele-taction) using a multi-functional sensory system, *IEEE Int. Conf. Robot. Autom.* (1993) pp. 955–960
- 30.99 D.G. Caldwell, S. Lawther, A. Wardle: Tactile perception and its application to the design of multi-modal cutaneous feedback systems, *IEEE Int. Conf. Robot. Autom.* (1996) pp. 3215–3221
- 30.100 C.G. Burdea: *Force and Touch Feedback for Virtual Reality* (Wiley Interscience, New York 1996)
- 30.101 M.C. Lin, M.A. Otaduy (Eds.): *Haptic Rendering: Foundations, Algorithms, and Applications* (AK Peters, Ltd., London 2008)
- 30.102 V. Hayward, K.E. MacLean: Do it yourself haptics, Part-I, *IEEE Robot. Autom. Mag.* **14**(4), 88–104 (2007)
- 30.103 K.E. MacLean, V. Hayward: Do It Yourself Haptics, Part-II. *IEEE Robot Autom Mag.*, to appear (2008 March issue)

- 30.104 V. Hayward, O.R. Astley, M. Cruz-Hernandez, D. Grant, G. Robles-De-La-Torre: Haptic interfaces and devices, *Sensor Rev.* **24**(1), 16–29 (2004)
- 30.105 K. Salisbury, F. Conti, F. Barbagli: Haptic rendering: introductory concepts, *IEEE Comput. Graphics Applicat.* **24**(2), 24–32 (2004)
- 30.106 V. Hayward, K.E. MacLean: A brief taxonomy of tactile illusions and demonstrations that can be done in a hardware store, *Brain Res. Bull.* (2007)
- 30.107 G. Robles-De-La-Torre: The importance of the sense of touch in virtual and real environments, *IEEE Multimedia* **13**(3), 24–30 (2006)

Telerobotics

31. Telerobotics

Günter Niemeyer, Carsten Preusche, Gerd Hirzinger

In this chapter we present an overview of the field of telerobotics with a focus on control aspects. Motivated by an historical perspective and some challenging applications of this research area a classification of control architectures is given, including an introduction to the different strategies. An emphasis is taken on bilateral control and force feedback, which is a vital research field today. Finally we suggest some literature for a closer engagement with the topic of telerobotics.

31.1 Overview	741	31.3 Control Architectures.....	746
31.2 Telerobotic Systems and Applications	743	31.3.1 Supervisory Control.....	746
31.2.1 Historical Perspective.....	743	31.3.2 Shared Control	748
31.2.2 Applications	744	31.3.3 Direct and Bilateral Teleoperation	749
		31.4 Bilateral Control and Force Feedback.....	751
		31.4.1 Position/Force Control	751
		31.4.2 Passivity and Stability.....	752
		31.4.3 Transparency and Multichannel Feedback	753
		31.4.4 Time Delay and Scattering Theory	754
		31.4.5 Wave Variables	754
		31.5 Conclusions and Further Reading	754
		References	755

31.1 Overview

Telerobotics is perhaps one of the earliest aspects of robotics. Literally meaning robotics at a distance, it is generally understood to refer to robotics with a human operator in control or human-in-the-loop. Any high-level, planning, or cognitive decisions are made by the human user, while the robot is responsible for their mechanical implementation. In essence, the *brain* is removed or distant from the *body*.

Herein the term *tele*, which is derived from the Greek and means distant, is generalized to imply a barrier between the user and the environment. This barrier is overcome by remote-controlling a robot at the environment, as indicated in Fig. 31.1. Besides distance, barriers may be imposed by hazardous environments or scaling to very large or small environments. All barriers have in common that the user cannot (or will not) physically reach the environment.

While the physical separation may be very small, with the human operator and the robot sometimes occupying the same room, telerobotic systems are often at

least conceptually split into two sites: the local site with the human operator and all elements necessary to support the system's connection with the user, which could be joysticks, monitors, keyboards, or other input/output devices, and the remote site, which contains the robot and supporting sensors and control elements.

To support this functionality, telerobotics integrates many areas of robotics. At the remote site, to operate the robot and execute the human's commands, the system may control the motion and/or forces of the robot. We refer to Chaps. 6 and 7 for detailed descriptions of these areas. Also, sensors are invaluable (Chap. 4), including force sensors (Chap. 19) and others (Part C). Meanwhile, at the local site information is often displayed haptically (Chap. 30).

A recent addition to telerobotics is the use of computer networks to transmit information between the two sites. This is the focus of Chap. 32 and opens up new possibilities in architectures. For example a single robot may be shared between multiple users or a single user may

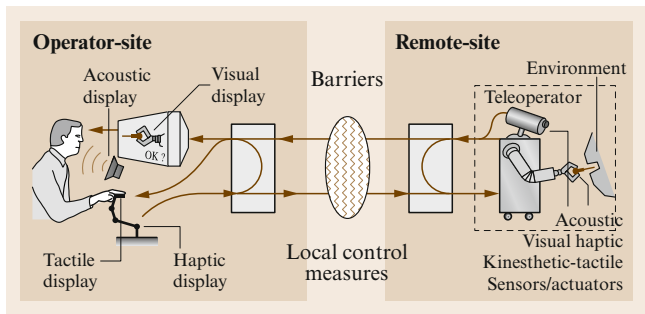


Fig. 31.1 Overview of a telerobotic system (from [31.1], adapted from [31.2])

control multiple robots. Web interfaces are simplifying this process, allowing access from anywhere on demand. In this chapter, we focus on point-to-point architectures with continual communications and operation.

We should also point out the relation between telerobotics and human exoskeletons, as described in Chap. 33. Exoskeletons are also controlled by a human operator, leaving all planning and high-level challenges to the user, and their control systems share many aspects with telerobotics. However, the two sites may be combined in an exoskeleton as the user directly touches and interacts with the robot. In this chapter, we will disallow any such connection.

The inclusion of the human operator makes telerobotics very attractive to handle unknown and unstructured environments. Applications are plentiful (Part F) and range from space robotics (Chap. 45) to dealing with hazardous environments (Chap. 48), from search and rescue situations (Chap. 50), to medical systems (Chap. 52) and rehabilitation (Chap. 53).

Before proceeding, we first define some basic terminology. Indeed many other terms are used nearly synonymously with telerobotics, in particular *teleoperation* and *telemanipulation*. Telerobotics is the most common, emphasizing a human's (remote) control of a robot. Teleoperation stresses the task-level operations, while telemanipulation highlights object-level manipulation.

Within telerobotics, a spectrum of control architectures has been used. *Direct control* or *manual control* falls at one extreme, indicating that the user is controlling the motion of the robot directly and without any automated help. At the other extreme, *supervisory control* implies that user's commands and feedback occur at a very high level and the robot requires substantial intelligence and/or autonomy. Between the two extrema

lie a variety of *shared control* architectures, where some degree of autonomy or automated help is available to assist the user.

In practice, many systems involve at least some level of direct control and include a joystick or similar device in the user interface, to accept the user's commands. As an instrumented mechanical device, this joystick can itself be viewed as a robot. The local and remote robots are called master and slave, respectively, while the system is referred to as a *master-slave system*. To provide direct control, the slave robot is programmed to follow the motions of the master robot, which is positioned by the user. It is not uncommon for the master robot (joystick) to be a kinematic replica of the slave, providing an intuitive interface.

Some master-slave systems provide force feedback, such that the master robot not only measures motions but also displays forces to the user. The user interface becomes fully bidirectional and such telerobotic systems are often called *bilateral*. The human-master interactions are a form of human-robot interaction (Chap. 57). The field of haptics (Chap. 30) also discusses bidirectional user interfaces, involving both motion and force, though more commonly to interface the user with virtual instead of remote environments. We should note that both motion and force may become the input or output to/from the user, depending on the system architecture.

Finally, *telepresence* is often discussed as an ultimate goal of master-slave systems and telerobotics in general. It promises to the user not only the ability to manipulate the remote environment, but also to perceive the environment as if encountered directly. The human operator is provided with enough feedback and sensations to feel present in the remote site. This combines the haptic modality with other modalities serving the human senses of vision, hearing or even smell and taste. We focus our descriptions on the haptic channel, which is created by the robotic hardware and its control systems. The master-slave system becomes the medium through which the user interacts with the remote environment and ideally they are fooled into forgetting about the medium itself. If this is achieved, we say that the master-slave system is transparent.

The chapter first examines telerobotic hardware and systems, before discussing various control architectures to operate these systems. A specific focus is placed on bilateral master-slave systems, which make the operator feel most connected to the remote environment and also present the largest stability and control problems.

31.2 Telerobotic Systems and Applications

Like mobile, industrial, and most areas of robotics, telerobotic systems are designed specifically with their tasks and requirements in mind. As such, many unique systems have evolved, of which we present an overview for different applications. We begin with a short historical perspective, then describe different applications with various robot designs and user interfaces.

31.2.1 Historical Perspective

Teleoperation enjoys a rich history and dates back to nuclear research by Raymond C. Goertz in the 1940s and 1950s. In particular, he created systems for humans to handle radioactive material from behind shielded walls. The first systems were electrical, controlled by an array of on-off switches to activate various motors and move various axes [31.3]. Without any feel, these manipulators were *slow and somewhat awkward to operate*, leading *Goertz* to build pairs of mechanically linked master-slave robots [31.3,4]. Connected by gears, linkages, and cables these systems allowed the operator to use natural hand motions and transmitted forces and vibrations through the connecting structure, yet they limited the distance between the operator and environment and required the use of kinematically identical devices, see Fig. 31.2. *Goertz* quickly recognized the value of electrically coupled manipulators and laid the foundations of modern telerobotics and bilateral force-reflecting positional servos [31.5].



Fig. 31.2 Raymond C. Goertz used electrical and mechanical teleoperators in the early 1950s to handle radioactive material. (Courtesy Argonne National Labs)

At the beginning of the 1960s the effects of time delay on teleoperation started to become a topic of research [31.6,7]. To cope with this problem the concept of supervisory control was introduced [31.2] and inspired the next years of development. In the late 1980s and early 1990s theoretical control came into play with Lyapunov-based analysis and network theory [31.8–13]. Using these new methods bilateral control of telerobotic systems became the vital research area it is today see Sect. 31.4. The growth of the Internet and its use as a communication medium fueled further this trend, adding the challenges of nondeterministic time delay.

On the hardware side, the Central Research Laboratory model M2 of 1982 was the first telerobotic system which realized force feedback while separating master and slave electronics. It was developed together with the Oak Ridge National Laboratory and was used for some time for a wide range of demonstration tasks including military, space or nuclear applications. The National Aeronautics and Space Administration (NASA) tested

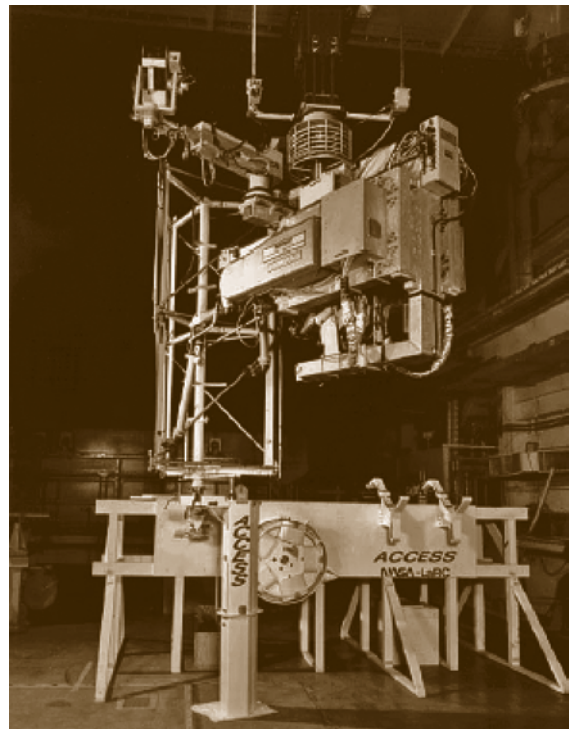


Fig. 31.3 The telerobotic system CRL Model M2 is used to verify the assembly of space truss structures (1982). (Courtesy of the Oak Ridge National Laboratory)

the M2 system to simulate the ACCESS space truss assembly with excellent results (Fig. 31.3). The advanced servomanipulator (ASM) was developed from the M2 to improve the remote maintainability of manipulators and intended as a foundation for telerobotic systems [31.14].

Also driven by the nuclear application, bilateral servomanipulators for teleoperation were developed in France in the Commission de Energie Atomique (CEA) by Vertut and his colleagues [31.15]. With the MA 23 they demonstrated telerobotic operation including computer-assisted functionalities to improve the operator's performance [31.16]. The assistance included software jigs and fixtures or virtual walls and restrictions [31.17], see also Sect. 31.3.2.

For space applications a dual-arm force reflecting telerobotic system was developed by Bejczy et al. at the Jet Propulsion Laboratory (JPL) [31.18]. In this approach for the first time kinematically and dynamically different master and slave systems were used, requiring control in Cartesian space coordinates. Figure 31.4 shows the master control station with its two back-drivable hand controllers. This system was used for simulating teleoperation in space.

In the 1980s and 1990s, as nuclear power activities began to decline, interest shifted to other areas such as space, medicine or undersea. Efforts were accelerated by the availability of increasing computer power as well as the introduction of novel hand controllers, e.g., the PHANTOM device [31.19], popularized by haptic applications in virtual reality (see Chap. 30).



Fig. 31.4 JPL ATOP control station (early 1980s). (JPL no. 19902Ac, courtesy of NASA/JPL-CALTECH)

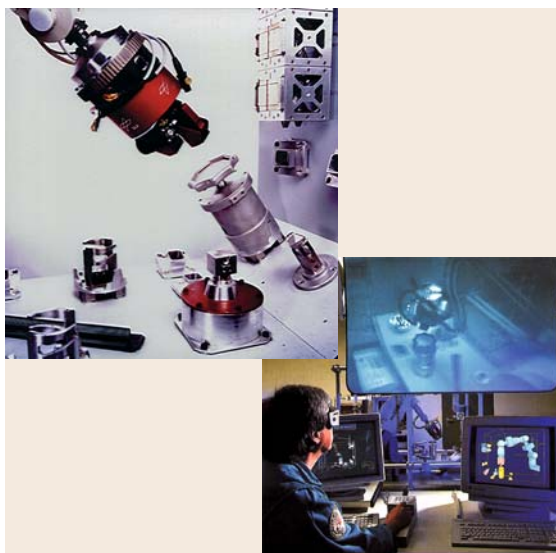


Fig. 31.5 ROTEX, the first remotely controlled robot in space (1993). Telerobot in space and ground operator station. (Courtesy of the German Aerospace Center, DLR)

In 1993 the first telerobotic system was flown in space with the German Spacelab Mission D2 on board the Space Shuttle Columbia. The robot technology experiment (ROTEX) demonstrated remote control of a space robot by means of local sensory feedback, predictive displays, and teleoperation [31.20]. In this experiment the round trip delay was 6–7 s, such that it was not feasible to include force feedback into the control loop.

With the first transatlantic telesurgery demonstration in 2001, Computer Motion demonstrated the feasibility of telerobotic systems even in the delicate field of surgery [31.21]. A surgeon in New York (USA) used a ZEUS system to perform a laparoscopic cholecystectomy on a patient located in Strasbourg (France), as depicted in Fig. 31.6. The system did not include force feedback, so the surgeon had to rely on visual feedback only.

In this perspective we gave only reference to the systems, that are seen to be milestones within the history of telerobotics. Several different systems have been developed and added value to the research field, which could not be mentioned here.

31.2.2 Applications

Telerobotic systems have been motivated by issues of human safety in hazardous environments (e.g., nuclear

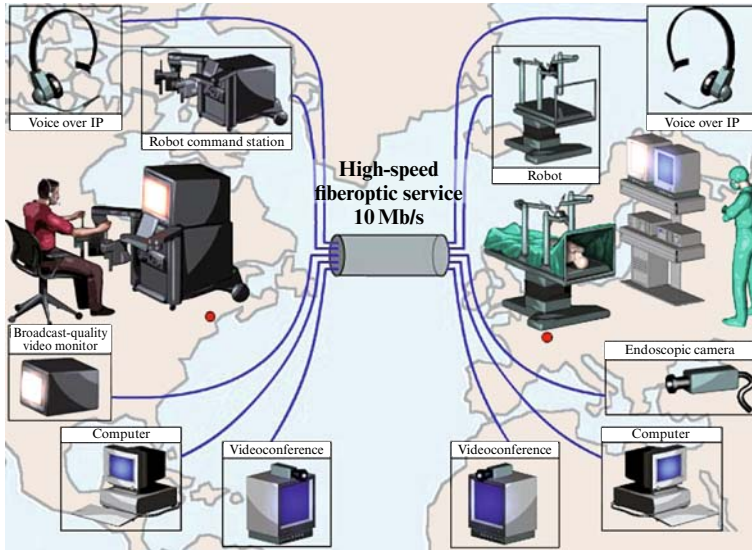


Fig. 31.6 Operation Lindberg. The first transcontinental telerobotic surgery (2001). (Courtesy M. Ghodoussi)

or chemical plants), the high cost of reaching remote environments (e.g., space), scale (e.g., power amplification or position scaling in micromanipulation or minimally invasive surgery), and many others. Not surprisingly, after their beginning in nuclear research, telerobotic systems have evolved to many fields of application. Nearly everywhere a robot is used, telerobotic systems can be found. The following are some of the more exciting uses.

In minimally invasive surgery telerobots allow procedures to be performed through small incisions, reducing the trauma to the patient compared to traditional surgery [31.22]. The da Vinci system, made by Intuitive Surgical Inc. [31.23] and shown in Fig. 31.7, is the only commercially available device at present. Other efforts,

however, include Computer Motion [31.24] and endoVia Medical [31.25] on the commercial side, as well as the University of Washington [31.26], Johns Hopkins University [31.27], the German Aerospace Center [31.28], and many others.



Fig. 31.7 Intuitive Surgical Inc. makes the da Vinci telerobotic system, which is used in minimally invasive surgery. (Courtesy ©2008 Intuitive Surgical, Inc.)



Fig. 31.8 tEODor, A telerobotic system for disarming of explosives. (Courtesy telerob Gesellschaft für Fernhantierungstechnik mbH, Ostfildern, Germany)

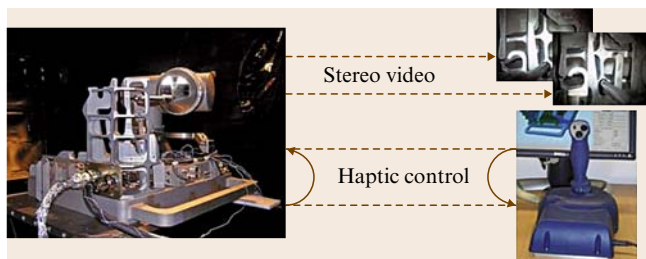


Fig. 31.9 ROKVISS, a telerobotic system providing stereo vision and haptic feedback to the ground operator. (Courtesy of the German Aerospace Center, DLR)

Protecting the operator from having to reach into a hazardous environment, telerobotic systems are widely used in nuclear or chemical industry. Some systems have been developed for the maintenance of high-voltage electrical power lines, which can be safely repaired without service interruption by a human operator using a telerobotic system. Disarming of explosives is another important task. Many systems like the telerob explosive ordnance disposal and observation robot (tEODor) shown in Fig. 31.8 or PackBot, made by iRobot [31.29], are used by police and military to disarm mines and other explosives. Similar vehicles are remote controlled for search and rescue in disaster zones [31.30].

31.3 Control Architectures

Compared to plain robotic systems, in which a robot executes a motion or other program without further consultation of a user or operator, telerobotic systems provide information to and require commands from the user. Their control architectures can be described by the style and level of this connection, as shown in Fig. 31.10. Organized in a spectrum, the three main categories are

- direct control
- shared control
- supervisory control

In practice, however, control architectures often include parts of all strategies.

Direct control implies no intelligence or autonomy in the system, so that all slave motion is directly controlled by the user via the master interface. If task execution is shared between direct control and local sensory feedback and autonomy, or if user feedback is augmented from virtual reality or other automatic aids, the architec-

ture is denoted as shared control. In supervisory control the user and slave are connected loosely with strong local autonomy, i. e., the operator is giving high-level commands, which are refined and executed by the telerobot. The following explains the architectures in reverse order, followed by a detailed treatment of direct and bilateral control in Sect. 31.3.3, which introduces the basic ideas for Sect. 31.4.

In orbital robotics the German technology experiment ROKVISS (robot component verification on ISS) is the most advanced telerobotic system [31.32]. Launched in 2004, it is installed outside the Russian module of the international space station. In this experiment advanced robot components of a slave system, including torque sensors and stereo video cameras, are validated in real space conditions. Due to a direct communication link between the space station and the operator station at DLR (German Aerospace Center), the time delay was reduced to about 20 ms allowing a bilateral control architecture with high-fidelity force feedback to the operator [31.33] (Fig. 31.9). This technology is leading toward robotic service satellites, called Robonauts, which can be remotely controlled from the ground to help real astronauts during extravehicular activity (EVA) or to perform repair and maintenance tasks [31.34].

ture is denoted as shared control. In supervisory control the user and slave are connected loosely with strong local autonomy, i. e., the operator is giving high-level commands, which are refined and executed by the telerobot. The following explains the architectures in reverse order, followed by a detailed treatment of direct and bilateral control in Sect. 31.3.3, which introduces the basic ideas for Sect. 31.4.

31.3.1 Supervisory Control

Supervisory control, introduced by *Ferrell* and *Sheridan* in 1967 [31.2], is derived from the analog of supervising a human subordinate staff member. The supervisor gives high-level directives to and receives summary information from, in this case, the robot. *Sheridan* describes this approach in comparison with manual and automatic robot control [31.35]: “Human operators are intermittently programming and continually receiving information from a computer that itself closes an au-

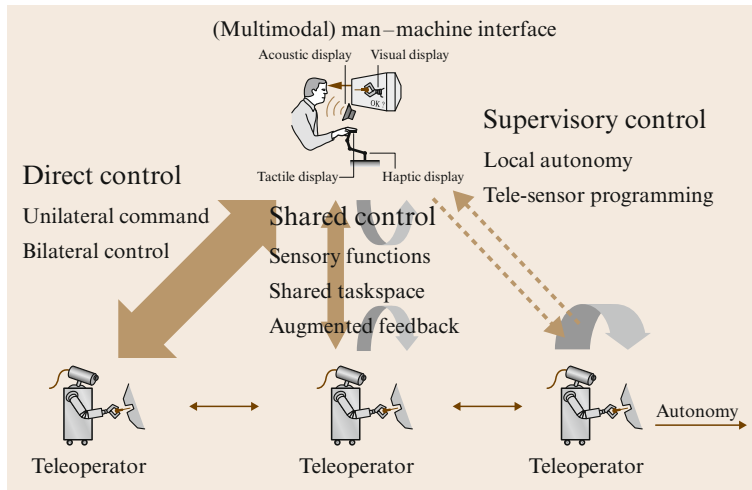


Fig. 31.10 Different concepts for telerobotic control architectures

tonomous control loop through artificial effectors and sensors.”

Today the autonomous control loops may be closed on the remote site, with only the state and model information being transmitted to the operator site. The operator supervises the telerobotic system and decides how to act and what to do. A special implementation of supervisory control is the telesensor programming approach, which is presented hereafter.

Telesensor Programming

The telesensor programming (TSP) approach is a shared autonomy concept that distributes intelligence between man and machine [31.36]. Presuming that sufficient information about the actual environment is available from the sensor systems, partial tasks can be executed independently on the machine level. Specifications and decisions on a high task planning level have to be done by human operators. Local sensory feedback loops are used by the robot system, while global task-level jobs have to be specified interactively by a human operator. This shared autonomy approach is the basis of the TSP paradigm, with which the robot can be teleprogrammed on a task directed level. The teaching of a robot system occurs not on the joint or cartesian manipulator level but on a higher language level, i. e., the operator plans activities at a level which can be performed by the robotic system independent of human intervention.

Figure 31.11 shows the structure of a TSP implementation, consisting of two control loops working in parallel. One loop controls the real (remote) system, which contains internal feedback for local autonomy. The other loop establishes a simulation environment

which is structurally equivalent to the real system, with a few exceptions.

Most importantly, any signal delay which may result from communication to the remote system, e.g., in space applications, is not duplicated in the simulation. This makes the simulation predictive with respect to the real system. A second exception is the display of internal variables in the simulation, which cannot be observed (measured) in the real system. This gives the operator or task planner more insight into what is happening or may happen in the system in response to commands. Communication between the two loops occurs via a common model data base which delivers a priori knowledge for execution on the remote system and a posteriori knowledge for model updating in the simulated world. For such a telerobotic control system unique tools are necessary to implement the required functionality. First a sophisticated simulation system has to be provided to emulate the real robot system. This includes the simulation of sensory perception within the real environment. Beyond this sensor simulation, the shared autonomy concept has to provide an efficient operator interface to setup task descriptions, to configure the task control parameters, to decide what kind of sensors and control algorithms should be used, and to debug an entire job execution phase.

In the field of telerobotics with large time delays, e.g., in space and undersea applications, this sensor-based task-directed programming approach [31.36] has advantages. Under direct visual feedback with a time delay of a few seconds, is not feasible for the human operator to handle the robot’s movements in a suitable way.

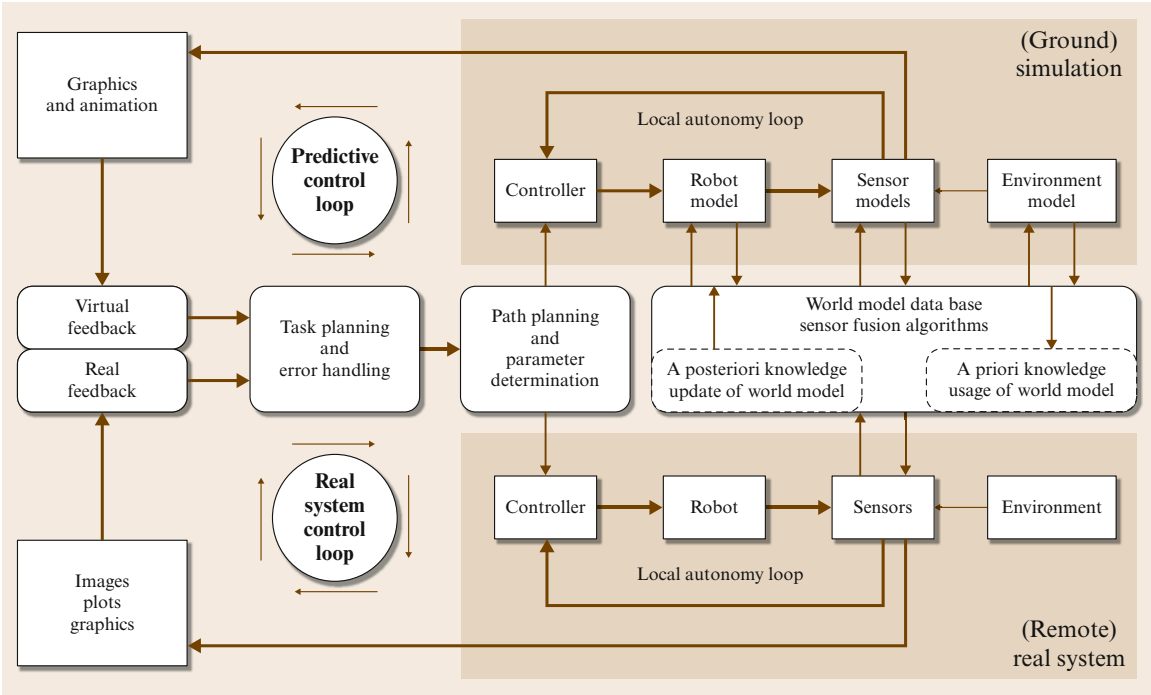


Fig. 31.11 The concept of telesensor programming as demonstrated during the ROTEX mission

Predictive simulation allows the operator to telemanipulate the remote system [31.37]. In addition, the use of force reflecting hand controllers to feed back force signals in shared and teleoperated control modes [31.38] or from the simulated predicted world can improve the operator's performance. Finally, an interactive supervisory user interface makes it possible to configure the environmental and control parameters.

The main feature of this telerobotic concept is to replace the time-delayed visual feedback with predictive stereo graphics including sensor simulation, providing a supervisory control technique that will allow a shift of more and more autonomy and intelligence to the robot system.

31.3.2 Shared Control

To enable telepresence in long-distance or risky applications (e.g., space or surgery) a sufficient *shared control* concept [31.39, 40] for the control of the teleoperator can be preferable to guarantee the safety of the teleoperator and/or task. Herein, shared control is based on local sensory feedback loops [31.41] at the teleoperator site (Fig. 31.11), by which gross commands were refined autonomously providing the teleoperator with a modest kind of sensory intelligence. The human operator originates gross path commands e.g. by using a kinesthetic feedback device, which are *fine-tuned* by the teleoperator.

In applications with large time delays the *shared autonomy* concept distributes intelligence between the operator and the teleoperator in the sense of a task-directed approach [31.42]. Control of the task is

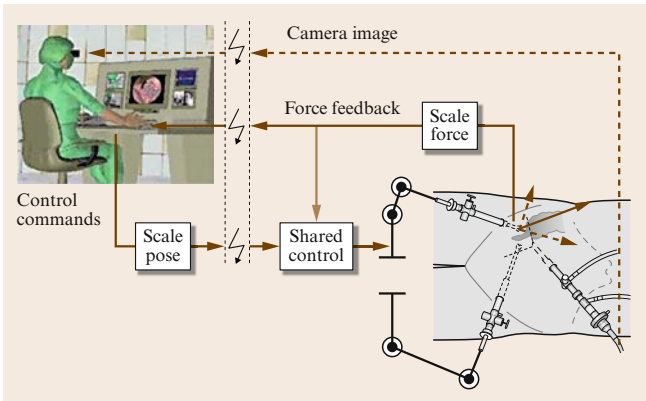


Fig. 31.12 An example for the shared control concept in telerobotic surgery

distributed between the human operator and the (autonomous) telerobot, such that each controls a subtask. An example can be seen in Fig. 31.12, where the autonomous part of the system controls and compensates the patient's movement, while the surgeon controls the operation itself on a virtual stabilized patient [31.43].

A special application of shared control is the use of *virtual fixtures* [31.44]. Virtual elements, such as virtual surfaces, guide tubes, or other appropriate objects, are superimposed into the visual and/or haptic scene for the user. These fixtures help the operator perform telerobotic or robot-assisted manipulation tasks by limiting movement into restricted regions and/or influencing movement along desired paths. Control is thus shared at the master site, taking advantage of preknowledge of the system or task to modify the user's commands and/or to combine them with autonomously generated signals.

Capitalizing on the accuracy of robotic systems, while sharing control with the operator, telerobotic systems with virtual fixtures can achieve safer and faster operation [31.45]. Abbott et al. describe the benefits by comparison to the common physical fixture of a ruler: "A straight line drawn by a human with the help of a ruler is drawn faster and straighter than a line drawn freehand. Similarly, a [master] robot can apply forces or positions to a human operator to help him or her draw a straight line." Based on the nature of the master robot and its controller, the virtual fixtures may indeed appear as impedance or admittance objects, i. e., providing corrective forces or positions to the user, respectively. In both cases, however, and in contrast to physical fixtures, the level and type of assistance can be programmed and varied.

31.3.3 Direct and Bilateral Teleoperation

To avoid difficulties in creating local autonomy, most telerobotic systems include some form of direct control: they allow the operator to specify the robot's motions. In the following we assume a master-slave system, i. e., the user is holding a joystick or master mechanism serving as an input device. We first describe unilateral operation, before focusing on bilateral control where the master also serves a display device.

Unilateral Acceleration or Rate Control

For underwater, airborne, or space applications, a slave robot may be a vehicle actuated by thrusters. Direct control thus requires the user to power the thrusters, which

in turn accelerate the vehicle. For other applications, the user may be required to command the rate or velocity of the vehicle or slave robot. In both scenarios, the input device is commonly a joystick, often spring centered, where the commands are proportional to the joystick displacement. For six-degree-of-freedom (DOF) applications, i. e., when the slave needs to be controlled in translation and orientation, a 6D-Space Mouse or alternatively often two joysticks are used for translation and orientation respectively.

Acceleration and rate control can require considerable effort for the operator to reach and hold a given target location. As expected, users can more accurately position a system under rate control than under acceleration control [31.46]. Indeed acceleration control necessitates users to regulate a second-order system versus a first-order system for rate control. Assuming the slave has local position feedback available, a control system is often incorporated locally, such that the user may specify position commands and is relieved from the dynamic control problem.

Position Control and Kinematic Coupling

Assuming that the slave is under position control, we next consider the kinematic coupling between master and slave, i. e., the mapping between master and slave positions. In particular, we must remember that the master mechanism moves in the master workspace, while the slave robot moves in the slave workspace. These two spaces are nearly always somewhat different.

Clutching and Offsets. Before even discussing how the two robots are coupled, we must understand that they are not always coupled, for example, before the system is turned on, master and slave robots may, for whatever reason, be placed in some initial position/configuration. We have three options of how to engage the system: (1) first autonomously move one or both robots so they come to the same position, (2) wait until someone (the user) externally moves one robot to match the location of the other, or (3) connect the two robots with some offset.

Once connected, most systems also allow a temporary disconnection between the two sites. The reason is twofold: to allow the user to rest without affecting the slave state and to allow a shift between the two robots. The latter is most important if the workspaces of both robots do not perfectly overlap. This is much like picking up your mouse off your mouse pad to reposition without moving the cursor.

In telerobotics the process is called *clutching* or sometimes also indexing. If clutching is allowed, or both robots are not constrained to start at the same location, the system must allow for offsets between the two robots.

Kinematically Similar Mechanisms. The simplest scenario involves a master and slave mechanism that are kinematically equivalent if not entirely identical. In this case, the two robots can be connected at a joint level. With \mathbf{q} denoting joint values and subscripts ‘m’ referring to the master, ‘s’ to the slave, ‘offset’ to a shared offset, and ‘d’ to a desired value, we can write

$$\begin{aligned}\mathbf{q}_{sd} &= \mathbf{q}_m + \mathbf{q}_{\text{offset}} , \\ \mathbf{q}_{md} &= \mathbf{q}_s - \mathbf{q}_{\text{offset}} .\end{aligned}\quad (31.1)$$

At the instance the two robots are to be connected or reconnected, the offset is computed as

$$\mathbf{q}_{\text{offset}} = \mathbf{q}_s - \mathbf{q}_m . \quad (31.2)$$

Most kinematically similar master–slave systems have the same workspace at both sites and do not allow clutching. By construction the offset is then always zero.

Depending on the controller architecture, the joint velocities may be similarly related, taking derivatives of (31.1). An offset in velocities is not necessary.

Kinematically Dissimilar Mechanisms. In many cases, the master and slave robots differ. Consider that the master is connected to the human user and thus should be designed accordingly. Meanwhile the slave works in some environment and may have a very different joint configuration and different number of joints. As a result, connecting the robots joint by joint may not be feasible or appropriate.

Instead kinematically dissimilar robots are connected at their tips. If \mathbf{x} is a robot’s tip position, we have

$$\begin{aligned}\mathbf{x}_{sd} &= \mathbf{x}_m + \mathbf{x}_{\text{offset}} , \\ \mathbf{x}_{md} &= \mathbf{x}_s - \mathbf{x}_{\text{offset}} .\end{aligned}\quad (31.3)$$

If orientations are also connected, with \mathbf{R} describing a rotation matrix, we have

$$\begin{aligned}\mathbf{R}_{sd} &= \mathbf{R}_m \mathbf{R}_{\text{offset}} , \\ \mathbf{R}_{md} &= \mathbf{R}_s \mathbf{R}_{\text{offset}}^\top ,\end{aligned}\quad (31.4)$$

where the orientational offset is defined as slave relative to master

$$\mathbf{R}_{\text{offset}} = \mathbf{R}_m^\top \mathbf{R}_s . \quad (31.5)$$

Again velocities and angular velocities may be connected if needed and do not require offsets.

Finally note that most telerobotic systems use a video camera at the remote site and a monitor at the local site. To make the connection appear natural, the slave position and orientation should be measured relative to the camera, while the master position and orientation should be measured relative to the user’s view.

Scaling and Workspace Mapping. Kinematically dissimilar master–slave robots are commonly also of different size. This means not only do they require clutching to fully map one workspace to another, but they often necessitate motion scaling. And so (31.3) becomes

$$\begin{aligned}\mathbf{x}_{sd} &= \mu \mathbf{x}_m + \mathbf{x}_{\text{offset}} , \\ \mathbf{x}_{md} &= \frac{(\mathbf{x}_s - \mathbf{x}_{\text{offset}})}{\mu} .\end{aligned}\quad (31.6)$$

The orientation, however, should not be scaled. The scale μ may be set to either map the two workspaces as best possible, or to provide the most comfort to the user.

If force feedback, introduced below, is provided, an equivalent force scale may be desired. This will prevent distortion of the remote environmental conditions, such as stiffness or damping, by the scaling.

Beyond linear scaling, several research efforts have created nonlinear or time-varying mappings, which deform the workspaces. These may effectively change the scale in the proximity of objects [31.47] or drift the offset to best utilize the master workspace [31.48].

Local Position Control. By construction we are now assuming that the slave follows a position command. This necessitates a local slave controller to regulate its position. In particular for kinematically dissimilar mechanisms, this will be a Cartesian tip position controller with appropriate inverse kinematics. For details we refer to Chap. 6.

If the slave robot has redundancies, these may be controlled either automatically to optimize some criterion or manually with additional user commands. We refer here to Chap. 11 for appropriate techniques.

31.4 Bilateral Control and Force Feedback

In pursuit of telepresence and to increase task performance, many master–slave systems incorporate force feedback. That is, the slave robot doubles as a sensor and the master functions as a display device, so that the system provides both forward and feedback pathways from the user to the environment and back. Figure 31.13 depicts the common architecture viewed as a chain of elements from the user to the environment.

The bilateral nature of this setup makes the control architecture particularly challenging: multiple feedback loops form and even without environment contact or user intervention, the two robots form an internal closed loop. The communications between the two sites often inserts delays into the system and this loop, so that stability of the system can be a challenging issue [31.49].

To present force information without stability problems, it is possible to use alternate displays, such as audio or tactile devices [31.50]. Meanwhile, the combination of vibrotactile methods with explicit force feedback can increase high-frequency sensations and provide benefits to the user [31.51]. Tactile shape sensing and display also extends the force information presented to the user [31.52].

In the following we discuss explicit force feedback. We first examine the basic architectures before discussing stability and some advanced techniques.

31.4.1 Position/Force Control

Two basic architectures couple the master and slave robots: position–position and position–force. We assume that the robot tips are to be connected by the equations of Sect. 31.3.3, giving the control laws for translation. Control of orientation or joint motions follows equivalent patterns.

Position–Position Architecture

In the simplest case, both robots are instructed to track each other. Both sites implement a tracking controller, often a proportional-derivative (PD) controller, to fulfill

these commands:

$$\begin{aligned} \mathbf{F}_m &= -\mathbf{K}_m(\mathbf{x}_m - \mathbf{x}_{md}) - \mathbf{B}_m(\dot{\mathbf{x}}_m - \dot{\mathbf{x}}_{md}), \\ \mathbf{F}_s &= -\mathbf{K}_s(\mathbf{x}_s - \mathbf{x}_{sd}) - \mathbf{B}_s(\dot{\mathbf{x}}_s - \dot{\mathbf{x}}_{sd}). \end{aligned} \quad (31.7)$$

If the position and velocity gains are the same ($\mathbf{K}_m = \mathbf{K}_s = \mathbf{K}$, $\mathbf{B}_m = \mathbf{B}_s = \mathbf{B}$), then the two forces are the same and the system effectively provides force feedback. This may also be interpreted as a spring and damper between the tips of each robot, as illustrated in Fig. 31.14. If the two robots are substantially different and require different position and velocity gains, the master–slave forces will be scaled and/or distorted.

Note we have assumed the slave is under impedance control and back-drivable. If the slave is admittance controlled, i. e., it accepts position commands directly, the second part of (31.7) is unnecessary.

Also note that by construction the user feels the slave’s controller forces, which include forces associated with the spring–damper and slave inertia in addition to environment forces. Indeed while moving without contact, the user will feel the inertial and other dynamic forces needed to move the slave. Furthermore, if the slave is not back-drivable, i. e., does not easily move under environment forces, the environment force may be entirely hidden from the user. Naturally this defeats the purpose of force feedback. In these cases, a local force control system may be used to render the slave back-drivable. Alternatively, a position–force architecture may be selected.

Position–Force Architecture

In the position–position architecture, the user is presented with the slave’s controller force. While this is very stable, it also means the user feels the friction and inertia in the slave robot, which the controller is actively driving to overcome. In many scenarios this is undesirable. To avoid the issue, position–force architectures place a force sensor at the tip of the slave robot and feedback the force from there. That is, the system is

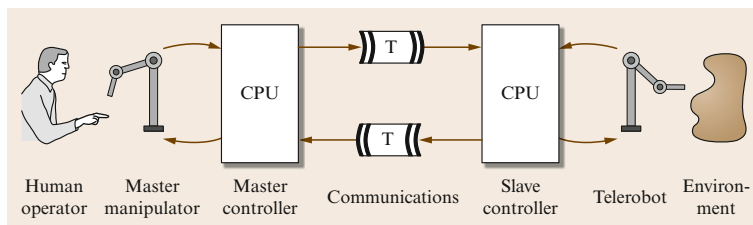


Fig. 31.13 A typical bilateral tele-operator can be viewed as a chain of elements reaching from user to environment

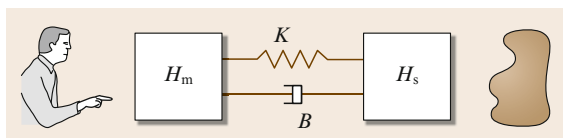


Fig. 31.14 A position–position architecture effectively creates a spring and damper between the two robots

controlled by

$$\begin{aligned} F_m &= F_{\text{sensor}} , \\ F_s &= -K_s(x_s - x_{sd}) - B_s(\dot{x}_s - \dot{x}_{sd}) . \end{aligned} \quad (31.8)$$

This allows the user to only feel the external forces acting between the slave and the environment and presents a more clear sense of the environment. However, this architecture is less stable: the control loop passes from master motion to slave motion to environment forces back to master forces. There may be some lag in the slave’s motion tracking not to mention any delay in communications. Meanwhile the loop gain can be very high: a small motion command can turn into a large force if the slave is pressing against a stiff environment. In combination, stability may be compromised in stiff contact and many systems exhibit contact instability in these cases.

31.4.2 Passivity and Stability

The two basic architectures presented in Sect. 31.4.1 clearly illustrate one of the basic tradeoffs and challenges in force feedback: stability and performance. Stability issues arise because any models of the system depend on the environment as well as the user. Both these elements are difficult to capture and, if we assume we want to explore unknown environments, impossible to predict. This issue makes a stability analysis very difficult. A common tool that avoids some of this issue is the concept of passivity. Although passivity provides only a sufficient (not a necessary) condition for stability, it incorporates the environmental uncertainty very well.

Passivity is an intuitive tool that examines the energy flows in a system and makes stability assertions if energy

is dissipated instead of generated. Three rules are of importance here. First, a system is passive if and only if it can not produce energy. That is the output energy from the system is limited by the initial and accumulated energy in the system. Second, two passive systems can be combined to form a new passive system. Third, the feedback connection of two passive systems is stable.

In the case of telerobotics, we generally assume that the slave robot will only interact with passive environments, that is, that the environments do not contain active motors or the like. Without the human operator, stability can therefore be assured if the system is also passive, without needing an explicit environment model.

On the master side the operator closes a loop and has to be considered in the stability analysis. In general, the master robot will be held by the user’s hand and arm. A variety of models and parameters describe the human arm dynamics, mainly in the form of a mass–damper–spring system. In [31.53] we find a summary of model parameters used by different authors. For an impedance-controlled haptic interface, common to most systems, the worst-case scenario for stability is the situation when the operator is not holding the haptic device [31.54, 55]. Thus we may elect to ignore the human operator in the analysis, assuming the human force equals zero ($F_{\text{human}} = 0$). A system then found to be stable will also be stable if the operator is interacting with the device.

To apply passivity, we take the system originally depicted in Fig. 31.13 and describe it as two-port elements in Fig. 31.15. We choose a sign convention, such that the power at every boundary is positive if flowing to the right. For example, at the first boundary, the positive power flow is the product of master velocity times applied (human) force

$$P_{\text{left}} = \dot{x}_m^T F_{\text{human}} . \quad (31.9)$$

Meanwhile at the last boundary, the positive power flow is the product of the slave velocity times the environment force (which ultimately opposes the human force)

$$P_{\text{right}} = \dot{x}_s^T F_{\text{env}} . \quad (31.10)$$



Fig. 31.15 A teleoperator can be analyzed as a chain-of-two port elements connecting the one-port operator to the one-port environment

Therefore the entire telerobotic system is passive if

$$\begin{aligned} \int_0^t P_{\text{input}} dt &= \int_0^t (P_{\text{left}} - P_{\text{right}}) dt \\ &= \int_0^t (\dot{\mathbf{x}}_m^\top \mathbf{F}_{\text{human}} - \dot{\mathbf{x}}_s^\top \mathbf{F}_{\text{env}}) dt \\ &> -E_{\text{store}}(0). \end{aligned} \quad (31.11)$$

To simplify the analysis, we can examine the passivity of each two-port element and then deduce the overall passivity. The master and slave robots are mechanical elements and hence passive. The controllers of a position–position architecture mimic a spring and damper, which are also passive elements. So without delay, a position–position architecture is passive.

While powerful to handle uncertainty, passivity can be overly conservative. Many controllers are overdamped if every subsystem is passive. In contrast, the combination of an active and a passive subsystem may be passive and stable and show less dissipation. This is particularly true for the cascaded arrangement of two-port elements in the telerobotic system of Fig. 31.15. From network theory, the Llewellyn criterion specifies when a possibly active two-port connected with any passive one-port becomes passive. This two-port is then labeled unconditionally stable, as it will be stable in connection to any two passive one-ports. The Llewellyn criterion may hence be used as a more general stability test for telerobotic systems or components [31.56].

Passive controllers are also limited as they cannot hide the dynamics of the slave robot. In the above position–position architecture, the user will feel the forces associated with the slave inertia. In contrast the position–force architecture hides the slave inertia and friction from the user. As such, when the user inserts kinetic energy into the master without feeling any resistance, the system itself creates and injects the kinetic energy for the slave. This violates passivity and provides another insight as to why the architecture suffers from potential stability problems.

31.4.3 Transparency and Multichannel Feedback

Both basic architectures can be captured by the general teleoperator control system described by Lawrence [31.13], and later expanded by Hashtrudi-Zaad and Salcudean [31.56] and shown in Fig. 31.16.

Knowing that ideally a teleoperator would have the master motion track the slave motion, as well as have the operator's force match the environment's force, both velocity (from which position may be integrated or vice versa) and force may be measured at both sites, for example, in this way the slave can start moving as soon as the user applies a force to the master, even before the master itself has moved.

Following these concepts derived in [31.13], we can examine the relationships between velocity and force, in the form of impedances and admittances. Note we do this in a single degree of freedom, assuming that all degrees of freedom may be treated independently. The environment will exhibit some impedance $Z_e(s)$ that is not known in advance and relates the environment force to the slave's velocity

$$\mathbf{F}_e(s) = Z_e(s) \mathbf{v}_s(s). \quad (31.12)$$

If we describe the teleoperator in whole as a two-port with a hybrid matrix formulation

$$\begin{pmatrix} \mathbf{F}_h(s) \\ \mathbf{v}_m(s) \end{pmatrix} = \begin{pmatrix} H_{11}(s) & H_{12}(s) \\ H_{21}(s) & H_{22}(s) \end{pmatrix} \begin{pmatrix} \mathbf{v}_s(s) \\ -\mathbf{F}_e(s) \end{pmatrix} \quad (31.13)$$

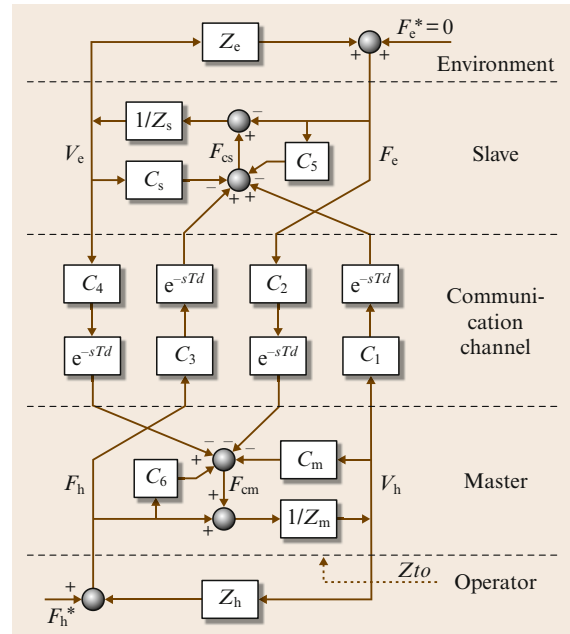


Fig. 31.16 In general, a controller will use both position and force information from both master and slave robot (after [31.56], adapted from [31.13])

then the user will perceive the impedance

$$Z_{to}(s) = \frac{F_h(s)}{v_m(s)} = (H_{11} - H_{12}Z_e)(H_{21} - H_{22}Z_e)^{-1}. \quad (31.14)$$

Transparency describes how close the user's perceived impedance comes to recreating the true environment impedance.

For a detailed treatment of passivity in telerobotics, impedance and admittance interpretations and designs, and transparency, we refer to some of the seminal works in [31.11–13, 56–59].

31.4.4 Time Delay and Scattering Theory

When delays occur in the communications between the local and remote site, even position–position architectures can suffer from serious instabilities [31.60, 61]. This can be traced to the communications block in Fig. 31.15, where the power entering the left side and exiting the right side do not add up. Rather energy may be generated inside the block, which feeds the instability [31.9].

Several approaches to operate under delay have been studied [31.62], in particular shared compliant control [31.63] and the addition of local force loops [31.64]. The use of the Internet for communication, adding variability to the delay, is also an area of interest [31.65, 66]. This further evokes issues of data reduction [31.67].

Here we note that natural wave phenomena are bilateral passive elements that tolerate delay. If the control system is described in the frequency domain and scattering matrices are used in place of impedance and admittance matrices, the system can tolerate delays [31.68]. Scattering matrices relate the sum of velocity and force to their difference, so that passivity becomes a condition on the system gain, which is unaffected by the delay. Alternatively, passivity may be explicitly observed and enforced to ensure stability [31.69–71].

31.4.5 Wave Variables

Building on the realization that delay communications can be active and that wave phenomena circumvent the

issue, wave variables provide an encoding scheme that is tolerant of delay [31.72]. Consider the power flowing through the system and separate the power moving forward and returning.

$$P = \dot{\mathbf{x}}^T \mathbf{F} = \frac{1}{2} \mathbf{u}^T \mathbf{u} - \frac{1}{2} \mathbf{v}^T \mathbf{v} = P_{\text{forward}} - P_{\text{return}}, \quad (31.15)$$

where the forward and returning power by construction have to be nonnegative. This leads to the definition of the wave variables

$$\begin{aligned} \mathbf{u} &= \frac{b\dot{\mathbf{x}} + \mathbf{F}}{\sqrt{2b}}, \\ \mathbf{v} &= \frac{b\dot{\mathbf{x}} - \mathbf{F}}{\sqrt{2b}}, \end{aligned} \quad (31.16)$$

where \mathbf{u} is the forward-moving and \mathbf{v} the returning wave.

If the normal signals are encoded into these wave variables, transmitted across the delay, and decomposed into regular variables, the system remains passive regardless of delay. In fact, in the wave domain, passivity corresponds to a wave gain of less than or equal to unity. No requirements are placed on phase and so lag does not destroy stability.

The wave impedance b relates velocity to force and provides a tuning knob to the operator. Large b values mean the system increases force feedback levels at the cost of feeling high inertial forces. Small values of b lower any unwanted sensations, making it easy to move quickly, but also lower the desirable environment forces. Ideally the operator would lower b when there is no risk of contact and raise b when contact is imminent.

Recent developments are incorporating both position–position and position–force architectures within the wave frame work, so the resulting system is stable with any environment, stable with any delay, yet maintains the feedback of high-frequency forces that help the operator identify happenings at the remote site [31.73]. To improve performance and assist the operator, predictors may also be incorporated [31.74].

31.5 Conclusions and Further Reading

Despite its age, telerobotics remains an exciting and vibrant area of robotics. In many ways, it forms a platform

which can utilize the advances in robotic technologies while simultaneously leveraging the proven skills and

capabilities of human users. Compare this, for example, with the development of the automobile and its relation to the driver. As cars are gradually becoming more sophisticated with added electronic stability control and navigation systems, they are becoming safer and more useful to their operators, not replacing them. Similarly telerobotics serves as a pathway for gradual progress and, as such, is perhaps best suited to fulfill robotics long-held promise of improving human life. It is seeing use in the challenging area of search and rescue. And with the recent developments and commercializations in telerobotic surgery systems, it is indeed impacting on the lives of tens of thousands of patients in a profound fashion and extending the reach of robotics into our world.

For further reading in the area of supervisory control, we refer to *Sheridan* [31.35]. Though published in 1992, it remains the most complete discussion on the topic. Unfortunately few other books are devoted to or even fully discuss telerobotics. In [31.75] many recent advances, including methods, experiments, applications, and developments, are collected. Beyond this, in the areas of bilateral and shared control, as well as to understand the various applications, we can only refer to the citations provided. Finally, in addition to the standard robotics journals, we note in particular *Presence: Teleoperators and Virtual Environments*, published by the MIT Press. Combined with virtual-reality applications, it focuses on technologies with a human operator.

References

- 31.1 M. Buss, G. Schmidt: Control problems in multi-modal telepresence systems, *Adv. Control, Highlights Eur. Control Conf. (ECC'99)* (1999) pp. 65–101
- 31.2 W.R. Ferrell, T.B. Sheridan: Supervisory control of remote manipulation, *IEEE Spectrum* **4**(10), 81–88 (1967)
- 31.3 R.C. Goertz: Fundamentals of general-purpose remote manipulators, *Nucleonics* **10**(11), 36–42 (1952)
- 31.4 R.C. Goertz: Mechanical master–slave manipulator, *Nucleonics* **12**(11), 45–46 (1954)
- 31.5 R.C. Goertz, F. Bevilacqua: A force-reflecting positional servomechanism, *Nucleonics* **10**(11), 43–45 (1952)
- 31.6 W.R. Ferrell: Remote manipulation with transmission delay, *IEEE Trans. Hum. Factors Electron.* **6**, 24–32 (1965)
- 31.7 T.B. Sheridan, W.R. Ferrell: Remote manipulative control with transmission delay, *IEEE Trans. Hum. Factors Electron.* **4**, 25–29 (1963)
- 31.8 F. Miyazaki, S. Matsubayashi, T. Yoshimi, S. Arimoto: A new control methodology towards advanced teleoperation of master–slave robot systems, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 3 (1986) pp. 997–1002
- 31.9 R.J. Anderson, M.W. Spong: Asymptotic stability for force reflecting teleoperators with time delay, *Int. J. Robot. Res.* **11**(2), 135–149 (1992)
- 31.10 G. Niemeyer, J.–J.E. Slotine: Stable adaptive teleoperation, *IEEE J. Oceanogr. Eng.* **16**(1), 152–162 (1991)
- 31.11 J.E. Colgate: Robust impedance shaping telemanipulation, *IEEE Trans. Robot. Autom.* **9**(4), 374–384 (1993)
- 31.12 B. Hannaford: A design framework for teleoperators with kinesthetic feedback, *IEEE Trans. Robot. Autom.* **5**(4), 426–434 (1989)
- 31.13 D.A. Lawrence: Stability and transparency in bilateral teleoperation, *IEEE Trans. Robot. Autom.* **9**(5), 624–637 (1993)
- 31.14 D. Kuban, H.L. Martin: An advanced remotely maintainable servomanipulator concept, *Proc. 1984 Natl. Top. Meet. Robot. Remote Handl. Hostile Environ.* (American Nuclear Society, Washington 1984)
- 31.15 J. Vertut, P. Coiffet: *Teleoperation and Robotics: Evolution and Development*, *Robot Technol.*, Vol. 3A (Hermes, Oslo 1985)
- 31.16 J. Vertut: MA23M contained servo manipulator with television camera, PICA and PIAD telescopic supports, with computer-integrated control, *Proc. 28th Remote Syst. Technol. Conf.*, Vol. 2 (1980) pp. 13–19
- 31.17 J. Vertut, P. Coiffet: Bilateral servo manipulator MA23 in direct mode and via optimized computer control, *Proc. 2nd RMS Conf.*, Vol. 12 (1977)
- 31.18 A.K. Bejczy: Towards advanced teleoperation in space. In: *Teleoperation and Robotics in Space*, *Prog. Astronaut. Aeronaut.*, Vol. 161, ed. by S.B. Skaar, C.F. Ruoff (American Institute of Aeronautics and Astronautics, Reston 1994) pp. 107–138
- 31.19 T.H. Massie, J.K. Salisbury: The phantom haptic interface: A device for probing virtual objects, *Proc. ASME Int. Mech. Eng. Congr. Exhib.* (Chicago 1994) pp. 295–302
- 31.20 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: Sensor-based space robotics – ROTEX and its telerobotic features, *IEEE Trans. Robot. Autom.* **9**(5), 649–663 (1993)
- 31.21 J. Marescaux, J. Leroy, F. Rubino, M. Vix, M. Simone, D. Mutter: Transcontinental robot assisted remote telesurgery: Feasibility and potential applications, *Ann. Surg.* **235**, 487–492 (2002)

- 31.22 G.H. Ballantyne: Robotic surgery, telerobotic surgery, telepresence, and telerobotics – Review of early clinical results, *Surg. Endosc.* **16**(10), 1389–1402 (2002)
- 31.23 G.S. Guthart, J.K. Salisbury: The IntuitiveTM telesurgery system: Overview and application, *Proc. IEEE Int. Conf. Robot. Autom.* (San Francisco 2000) pp. 618–621
- 31.24 J.M. Sackier, Y. Wang: Robotically assisted laparoscopic surgery: From concept to development. In: *Computer-Integrated Surgery: Technology and Clinical Applications*, ed. by R.H. Taylor, S. Laval-lée, G.C. Burdea, R. Mösges (MIT Press, Cambridge 1996) pp. 577–580, Chap. 45
- 31.25 D.H. Birkett: Electromechanical instruments for endoscopic surgery, *Minimally Invasive Therapy and Allied Technologies* **10**(6), 271–274 (2001)
- 31.26 J. Rosen, B. Hannaford: Doc at a distance, *IEEE Spectrum* **8**(10), 34–39 (2006)
- 31.27 A.M. Okamura: Methods for haptic feedback in teleoperated robot-assisted surgery, *Industr. Robot* **31**(6), 499–508 (2004)
- 31.28 T. Ortmaier, B. Deml, B. Kübler, G. Passig, D. Reintsema, U. Seibold: Robot assisted force feedback surgery. In: *Advances in Telerobotics*, Springer Tracts Adv. Robot., Vol. 31, ed. by M. Ferre, M. Buss, R. Aracil, C. Melchiorri, C. Balaguer (Springer, Berlin, Heidelberg 2007) pp. 361–79, Chap. 21
- 31.29 B.M. Yamauchi: PackBot: A versatile platform for military robotics, *Proc. SPIE* **5422**, 228–237 (2004)
- 31.30 R.R. Murphy: Trial by fire [rescue robots], *IEEE Robot. Autom. Mag.* **11**(3), 50–61 (2004)
- 31.31 J. Wright, A. Trebi-Ollennu, F. Hartman, B. Cooper, S. Maxwell, J. Yen, J. Morrison: Driving a Rover on Mars Using the Rover Sequencing and Visualization Program, *International Conference on Instrumentation, Control and Information Technology* (Okayama University, Okayama 2005)
- 31.32 G. Hirzinger, K. Landzettel, D. Reintsema, C. Preusche, A. Albu-Schäffer, B. Rebele, M. Turk: ROKVISS – Robotics component verification on ISS, *Proc. 8th Int. Symp. Artif. Intell. Robot. Autom. Space (ISAIRAS)* (Munich 2005) p. Session2B
- 31.33 C. Preusche, D. Reintsema, K. Landzettel, G. Hirzinger: ROKVISS – Preliminary results for telepresence mode, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)* (Peking 2006) pp. 4595–4601
- 31.34 G. Hirzinger, K. Landzettel, B. Brunner, M. Fischer, C. Preusche, D. Reintsema, A. Albu-Schäffer, G. Schreiber, M. Steinmetz: DLR's robotics technologies for on-orbit servicing, *Adv. Robot. – Special Issue Service Robots in Space* **18**(2), 139–174 (2004)
- 31.35 T.B. Sheridan: *Telerobotics, Automation and Human Supervisory Control* (MIT Press, Cambridge 1992)
- 31.36 G. Hirzinger, J. Heindl, K. Landzettel, B. Brunner: Multisensory shared autonomy – a key issue in the space robot technology experiment ROTEX, *Proc. RSJ/IEEE Int. Conf. Intell. Robot. Syst.* (1992)
- 31.37 A.K. Bejczy, W.S. Kim: Predictive displays and shared compliance control for time-delayed tele-manipulation, *IEEE Int. Workshop Intell. Robot. Syst.* (Ibaraki 1990) pp. 407–412
- 31.38 P. Backes, K. Tso: UMI: An interactive supervisory and shared control system for telerobotics, *Proc. IEEE Int. Conf. Robot. Autom.*, Cincinnati (1990) pp. 1096–1101
- 31.39 L. Conway, R. Volz, M. Walker: Tele-autonomous systems: Methods and architectures for intermingling autonomous and telerobotic technology, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 2 (Raleigh 1987) pp. 1121–1130
- 31.40 S. Hayati, S.T. Venkataraman: Design and implementation of a robot control system with traded and shared control capability, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 3 (Scottsdale 1989) pp. 1310–1315
- 31.41 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: ROTEX – The first remotely controlled robot in space, *Proc. IEEE Int. Conf. Robot. Autom.*, Vol. 3 (San Diego 1994) pp. 2604–2611
- 31.42 B. Brunner, K. Arbter, G. Hirzinger: Task directed programming of sensor based robots, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Vol. 2 (Munich 1994) pp. 1080–1087
- 31.43 T. Ortmaier, M. Gröger, D.H. Boehm, V. Falk, G. Hirzinger: Motion estimation in beating heart surgery, *IEEE Transactions on Biomedical Engineering (TBME)* **52**(10), 1729–1740 (2005)
- 31.44 L. Rosenberg: Virtual fixtures: Perceptual tools for telerobotic manipulation, *Proc. IEEE Virtual Real. Int. Symp.*, New York (Seattle 1993) pp. 76–82
- 31.45 J.J. Abbott, P. Marayong, A.M. Okamura: Haptic virtual fixtures for robot-assisted manipulation, *Proc. 12th Int. Symp. Robot. Res.*, Vol. 28 (2007) pp. 49–64
- 31.46 M.J. Massimino, T.B. Sheridan, J.B. Roseborough: One handed tracking in six degrees of freedom, *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Vol. 2 (Cambridge 1989) pp. 498–503
- 31.47 A. Casals, L. Munoz, J. Amat: Workspace deformation based teleoperation for the increase of movement precision, *Proc. IEEE Int. Conf. Robot. Autom.* (Taipei 2003) pp. 2824–2829
- 31.48 F. Conti, O. Khatib: Spanning large workspaces using small haptic devices, *Proc. 1st Joint Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleoper. Syst.* (Pisa 2005) pp. 183–188

- 31.49 R.W. Daniel, P.R. McAree: Fundamental limits of performance for force reflecting teleoperation, *Int. J. Robot. Res.* **17**(8), 811–830 (1998)
- 31.50 M.J. Massimino, T.B. Sheridan: Sensory substitution for force feedback in teleoperation, *Presence Teleoper. Virtual Environ.* **2**(4), 344–352 (1993)
- 31.51 D.A. Kontarinis, R.D. Howe: Tactile display of vibratory information in teleoperation and virtual environments, *Presence Teleoper. Virtual Environ.* **4**(4), 387–402 (1995)
- 31.52 D.A. Kontarinis, J.S. Son, W.J. Peine, R.D. Howe: A tactile shape sensing and display system for teleoperated manipulation, *Proc. IEEE Int. Conf. Robot. Autom.* (Nagoya 1995) pp. 641–646
- 31.53 J.J. Gil, A. Avello, Á. Rubio, J. Flórez: Stability analysis of a 1 DOF haptic interface using the Routh–Hurwitz criterion, *IEEE Trans. Control Syst. Technol.* **12**(4), 583–588 (2004)
- 31.54 N. Hogan: Controlling impedance at the man/machine interface, *Proc. IEEE Int. Conf. Robot. Autom.* (Scottsdale 1989) pp. 1626–1631
- 31.55 R.J. Adams, B. Hannaford: Stable haptic interaction with virtual environments, *IEEE Trans. Robot. Autom.* **15**(3), 465–474 (1999)
- 31.56 K. Hashtrudi-Zaad, S.E. Salcudean: Analysis of control architectures for teleoperation systems with impedance/admittance master and slave manipulators, *Int. J. Robot. Res.* **20**(6), 419–445 (2001)
- 31.57 Y. Yokokohji, T. Yoshikawa: Bilateral control of master–slave manipulators for ideal kinesthetic coupling – formulation and experiment, *IEEE Trans. Robot. Autom.* **10**(5), 605–620 (1994)
- 31.58 K.B. Fite, J.E. Speich, M. Goldfarb: Transparency and stability robustness in two-channel bilateral telemanipulation, *ASME J. Dyn. Syst. Meas. Control* **123**(3), 400–407 (2001)
- 31.59 S.E. Salcudean, M. Zhu, W.-H. Zhu, K. Hashtrudi-Zaad: Transparent bilateral teleoperation under position and rate control, *Int. J. Robot. Res.* **19**(12), 1185–1202 (2000)
- 31.60 W.R. Ferrell: Remote manipulation with transmission delay, *IEEE Trans. Hum. Factors Electron.* **6**, 24–32 (1965)
- 31.61 T.B. Sheridan: Space teleoperation through time delay: Review and prognosis, *IEEE Trans. Robot. Autom.* **9**(5), 592–606 (1993)
- 31.62 A. Eusebi, C. Melchiorri: Force reflecting telemanipulators with time-delay: Stability analysis and control design, *IEEE Trans. Robot. Autom.* **14**(4), 635–640 (1998)
- 31.63 W.S. Kim, B. Hannaford, A.K. Bejczy: Force-reflection and shared compliant control in operating telemanipulators with time delays, *IEEE Trans. Robot. Autom.* **8**(2), 176–185 (1992)
- 31.64 K. Hashtrudi-Zaad, S.E. Salcudean: Transparency in time-delayed systems and the effect of local force feedback for transparent teleoperation, *IEEE Trans. Robot. Autom.* **18**(1), 108–114 (2002)
- 31.65 R. Oboe, P. Fiorini: A design and control environment for internet-based telerobotics, *Int. J. Robot. Res.* **17**(4), 433–449 (1998)
- 31.66 S. Munir, W.J. Book: Control techniques and programming issues for time delayed internet based teleoperation, *J. Dyn. Syst. Meas. Control* **125**(2), 205–214 (2003)
- 31.67 S. Hirche, M. Buss: Transparent data reduction in networked telepresence and teleaction systems. Part II: Time-delayed communication, *Presence Teleoper. Virtual Environ.* **16**(5), 532–542 (2007)
- 31.68 S. Stramigioli, A. van der Schaft, B. Maschke, C. Melchiorri: Geometric scattering in robotic telemanipulation, *IEEE Trans. Robot. Autom.* **18**(4), 588–596 (2002)
- 31.69 B. Hannaford, J.H. Ryu: Time domain passivity control of haptic interfaces, *IEEE Trans. Robot. Autom.* **18**(1), 1–10 (2002)
- 31.70 J.-H. Ryu, C. Preusche, B. Hannaford, G. Hirzinger: Time domain passivity control with reference energy following, *IEEE Trans. Control Syst. Technol.* **13**(5), 737–742 (2005)
- 31.71 J. Artigas, C. Preusche, G. Hirzinger: Time domain passivity-based telepresence with time delay, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, Peking (Peking 2006) pp. 4205–4210
- 31.72 G. Niemeyer, J.-J.E. Slotine: Telemanipulation with time delays, *Int. J. Robot. Res.* **23**(9), 873–890 (2004)
- 31.73 N.A. Tanner, G. Niemeyer: High-frequency acceleration feedback in wave variable telerobotics, *IEEE/ASME Trans. Mechatron.* **11**(2), 119–127 (2006)
- 31.74 S. Munir, W.J. Book: Internet-based teleoperation using wave variables with prediction, *IEEE/ASME Trans. Mechatron.* **7**(2), 124–133 (2002)
- 31.75 M. Ferre, M. Buss, R. Aracil, C. Melchiorri, C. Balague (Eds.): *Advances in Telerobotics*, Springer Tracts Adv. Robot. (Springer, Berlin, Heidelberg 2007)

Networked T

32. Networked Telerobots

Dezhen Song, Ken Goldberg, Nak Young Chong

Telerobots, remotely controlled robots, are widely used to explore undersea terrains and outer space, to defuse bombs, and to clean up hazardous waste. Until 1994, telerobots were accessible only to trained and trusted experts through dedicated communication channels. This chapter describes networked telerobots, a new class of telerobots controllable over networks such as the Internet, that are accessible to the general public. This chapter will describe relevant network technology, the history of networked telerobots within the broader field of teleoperation, properties of networked telerobots, how to build a networked robot, example systems, and topics for future research.

32.1	Overview and Background	759
32.2	A Brief History	760
32.3	Communications and Networking	761
	32.3.1 The Internet	762
	32.3.2 Wired Communication Links	763
	32.3.3 Wireless Links	763
	32.3.4 Properties of Networked Telerobotics	764
	32.3.5 Building a Networked Telerobotic System	764
	32.3.6 State-Command Presentation	765
	32.3.7 Command Execution/State Generation	767
	32.3.8 Collaborative Control	768
32.4	Conclusion and Future Directions	769
	References	769

32.1 Overview and Background

As illustrated in Fig. 32.1, the broader field of teleoperation, where the primary concerns are stability and time delay, is covered in Chap. 31. The field of networked *robots*, where autonomous robots and sensors communicate over local networks, is covered in Chap. 41. Networked *telerobots*, the subject of the present chapter, focuses on teleoperated robot systems that are accessible by the public via web browsers.

By 2006, several hundred networked telerobots have been developed and put online for public use. Many papers have been published describing these systems and a book on this subject by Goldberg and Siegwart is available [32.1]. Updated information about new research and an archive/survey of networked telerobots is available on the website of the IEEE technical committee on networked robots, which fosters research in both networked telerobots and networked robots (IEEE Technical Committee on Networked Robots <http://tab.ieee-ras.org/>).

Networked telerobots have the following properties

- The physical world is affected by a device that is locally controlled by a network *server*, which communicates with remote human users through web browsers such as Internet Explorer or Firefox, which are generally referred to as *clients*. As of 2006, the standard protocol for network browsers is the hypertext transfer protocol (HTTP), a stateless transmission protocol.
- Most networked telerobots are continuously accessible (online), 24 hours a day, 7 days a week.
- Since hundreds of millions of people now have access to the Internet, mechanisms are needed to handle client authentication and contention.
- Input and output for networked telerobots is usually achieved with the standard computer screen, mouse, and keyboard.
- Clients may be inexperienced or malicious, so online tutorials and safeguards are generally required.

32.2 A Brief History

Like many technologies, remotely controlled devices were first imagined in science fiction. In 1898, *Nicola Tesla* [32.2] demonstrated a radio-controlled boat in New York's Madison Square Garden. The first major experiments in teleoperation were motivated by the need to handle radioactive materials in the 1940s. *Gortz* demonstrated one of the first bilateral simulators in the 1950s at the Argonne National Laboratory [32.3]. Remotely operated mechanisms have been designed for use in inhospitable environments such as under-sea [32.4] and space exploration [32.5]. At General Electric, *Mosher* [32.6] developed a two-arm teleoperator with video cameras. Prosthetic hands were also applied to teleoperation [32.7]. More recently, teleoperation has been considered for medical diagnosis [32.8], manufacturing [32.9], and micromanipulation [32.10]. See Chap. 31 and the book from *Sheridan* [32.11] for excellent reviews on teleoperation and telerobotics research.

The concept of hypertext (linked references) was proposed by Vannevar Bush in 1945 and was made possible by subsequent developments in computing and networking. In the early 1990s, Berners-Lee introduced the HTTP. A group of students led by Marc Andreessen developed an open-source version of the first graphical user interface, the *Mosaic* browser, and put it online in 1993. The first networked camera, or *webcam*, went online in November 1993 [32.12].

Approximately nine months later, the first networked telerobot went online. K. Goldberg and M. Mascha's Mercury Project combined an IBM industrial robot arm with a digital camera and used the robot's air nozzle to allow remote users to excavate for buried artifacts in a sandbox [32.13, 14]. Working independently, a team led by K. Taylor and J. Trevelyan at the University of Western Australia demonstrated a remotely controlled six-axis telerobot in September 1994 [32.15, 16]. These early projects pioneered a new field of networked telerobots. See [32.17–25] for other examples. An online archive of Networked Telerobots is at: <http://ford.ieor.berkeley.edu/ir/>.

Networked telerobots are a special case of *supervisory control* telerobots, as proposed by *Sheridan* and colleagues [32.11]. Under supervisory control, a local computer plays an active role in closing the feedback loop. Most networked robotics are type c supervisory control systems (see Fig. 32.2).

Although the majority of networked telerobotic systems consist of a single human operator and a single robot [32.26–33], *Chong et al.* [32.34] propose a useful taxonomy: single operator single robot (*SOSR*), single operator multiple robot (*SOMR*), multiple operator single robot (*MOSR*), and multiple operator multiple robot (*MOMR*).

The decade from 1995–2005 witnessed extensive development in networked telerobots. New systems, new experiments, and new applications now go well beyond the traditional fields such as defense, space, and nuclear material handling [32.11] that motivated teleoperation in early 1950s. As the Internet introduces universal access to every corner of life, the impact of networked robots becomes broader and deeper in modern society. Recent applications range from education, industry, commercial, health care, geology, and environmental monitoring, to entertainment and arts.

Networked telerobots provide a new medium for people to interact with a remote environment. A networked robot can provide more interactivity beyond that provided by a normal videoconferencing system. The physical robot not only represents the remote person but also transmits multimodal feedback to the person, which is often referred as *telepresence* in the literature [32.29]. *Paulos* and *Canny's* personal roving presence (PRoP) robot [32.35] and *Jouppi* and *Thomas's* surrogate robot [32.29] are recent representative work.

Networked telerobots have great potential for education and training. In fact, one of the earliest networked telerobot systems [32.36] originated from the idea of a remote laboratory. Networked telerobots provide universal access to the general public, who may have little to no knowledge of robots, with opportunities to understand, learn, and operate robots, which were expensive scientific equipment limited to universities and large corporate laboratories before. Built on networked telerobots, online remote laboratories [32.37, 38] greatly improve distance learning by providing an interactive experience; for example, teleoperated telescopes help students to understand astronomy [32.39]. Teleoperated microscopes [32.40] help student to observe microorganisms. The Tele-Actor project [32.41] allows a group of students to remotely control a human tele-actor to visit environments that are normally not accessible to them such as cleanroom environments for a semiconductor manufactory facility and DNA analysis laboratories.

32.3 Communications and Networking

Below is a short review of relevant terminologies and technologies on networking. For details, see the texts by [32.42].

A communication network includes three elements: *links*, *routers/switchers*, and *hosts*. Links refer to the physical medium that carry bits from one place to an-

other. Examples of links include copper or fiber-optic cables and wireless (radio frequency or infrared) channels. Switches and routers are hubs that direct digital information between links. Hosts are communication end points such as browsers, computers, and robots.

Networks can be based in one physical area (local-area network, or **LAN**), or distributed over wide distances (wide-area network, or **WAN**). Access control is a fundamental problem in networking. Among a variety of methods, the *ethernet* protocol is the most popular. Ethernet provides a broadcast-capable multi-access **LAN**. It adopts a carrier-sense multiple-access (**CSMA**) strategy to address the multiple-access problem. Defined in the **IEEE 802.x** standard, **CSMA** allows each host to send information over the link at any time. Therefore, collisions may happen between two or more simultaneous transmission requests. Collisions can be detected either by directly sensing the voltage in the case of wired networks, which is referred to as collision detection (**CSMA/CD**), or by checking the time-out of

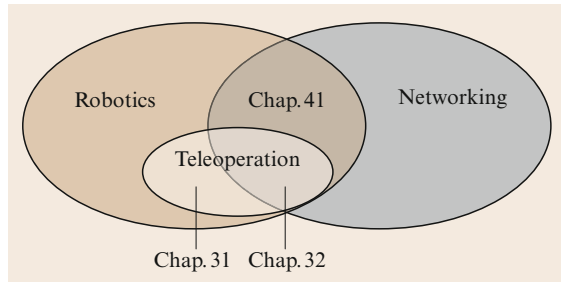


Fig. 32.1 Relationship between the subjects of networked telerobots (Chap. 32, the present chapter), teleoperation (Chap. 31), and networked robots (Chap. 41)

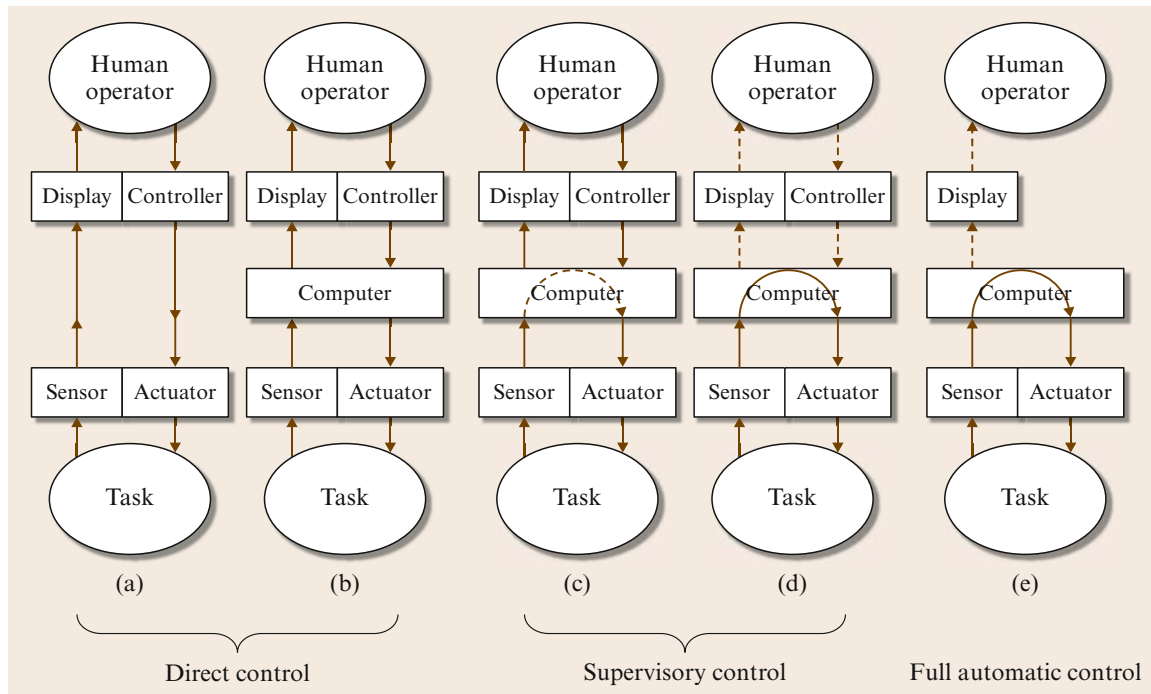


Fig. 32.2 A spectrum of teleoperation control modes adapted from *Sheridan's* text [32.11]. We label them (a–e), in order of increasing robot autonomy. At the *far left* would be a mechanical linkage where the human directly operates the robot from another room through sliding mechanical bars, and on the *far right* is the system where the human role is limited to observation/monitoring. In (c–e), the *dashed lines* indicated that communication may be intermittent

an anticipated acknowledgement in wireless networks, which is referred to as collision avoidance (CSMA/CA). If a collision is detected, both/all senders randomly back off a short period of time before retransmitting. CSMA has a number of important properties: (1) it is a completely decentralized approach, (2) it does not need clock synchronization over the entire network, and (3) it is very easy to implement. However, the disadvantages of CSMA are: (1) the efficiency of the network is not very high and (2) the transmission delay can change drastically.

As mentioned previously, LANs are interconnected with each other via routers/switchers. The information transmitted is in packet format. A packet is a string of bits and usually contains the source address, the destination address, content bits, and a checksum. Routers/switchers distribute packets according to their routing table. Routers/switchers have no memory of packets, which ensures scalability of the network. Packets are usually routed according to a first-in first-out (FIFO) rule, which is independent of the application. The packet formats and addresses are independent of the host technology, which ensures extensibility. This routing mechanism is referred to as packet switching in the networking literature. It is quite different from a traditional telephone network, which is referred to as circuit switching. A telephone network is designed to guarantee a dedicated circuit between a sender and a receiver once a phone call is established. The dedicated circuitry ensures communication quality. However, it requires a large number of circuits to ensure the quality of service (QoS), which leads to poor utilization of the overall network. A packet-switching network cannot guarantee dedicated bandwidth for each individual pair of transmissions, but it improves overall resource utilization. The Internet, which is the most popular communication media and the infrastructure of networked telerobots, is a packet-switching network.

32.3.1 The Internet

The creation of the Internet can be traced back to US Department of Defense’s (DoD) APRA NET network in the 1960s. There are two features of the APRA NET network that enabled the successful evolution of the Internet. One feature is the ability for information (packets) to be rerouted around failures. Originally this was designed to ensure communication in the event of a nuclear war. Interestingly, this dynamic routing capability also allows the topology of the Internet to grow easily. The second important feature is the ability for heterogeneous net-

works to interconnect with one another. Heterogeneous networks, such as X.25, G.701, ethernet, can all connect to the Internet as long as they can implement the Internet protocol (IP). The IP is media, operating system (OS), and data rate independent. This flexible design allows a variety of applications and hosts to connect to the Internet as long as they can generate and understand IP.

Figure 32.3 illustrates a four-layer model of the protocols used in the Internet. On the top of the IP, we have two primary transport layer protocols: the transmission control protocol (TCP) and the user data protocol (UDP). TCP is an end-to-end transmission control protocol. It manages packet ordering, error control, rate control, and flow control based on packet round-trip time. TCP guarantees the arrival of each packet. However, excessive retransmission of TCP in a congested network may introduce undesirable time delays in a networked telerobotic system. UDP behaves differently; it is a broadcast-capable protocol and does not have a retransmission mechanism. Users must take care of error control and rate control themselves. UDP has a lot less overhead compared to TCP. UDP packets are transmitted at the sender’s preset rate and the rate is changed based on the congestion of a network. UDP has great potential, but it is often blocked by firewalls because of a lack of a rate control mechanism. It is also worth mentioning that the widely accepted term TCP/IP refers to the family of protocols that build on IP, TCP, and UDP.

In the application layer of the Internet protocols, the HTTP is one of the most important protocols. HTTP is the protocol for the World Wide Web (WWW). It allows the sharing of multimedia information among

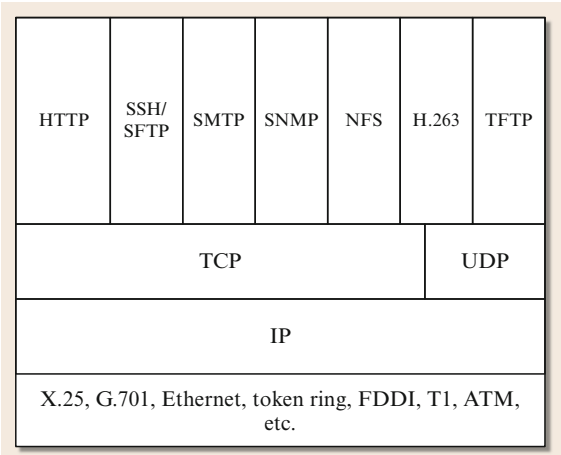


Fig. 32.3 A four-layer model of internet protocols (after [32.42])

Table 32.1 Last-mile Internet speed by wired connection type. If not specified, the downstream transmission and the upstream transmission share the same bandwidth

Types	Bits per second
Modem (V.92)	Up to 56 K
Integrated Services Digital Network (ISDN) BRI	64–128 K
High Data Rate Digital Subscriber Line (HDSL)	1.544 M duplex on two twisted-pair lines
Assymetric Digital Subscriber Line (ADSL)	1.544–6.1 M downstream, 16–640 K upstream
Cable modem	2–4 M downstream, 400–600 K upstream
Fiber to the home (FTTH)	5–30 M downstream, 2–5 M upstream
Internet II/III node	≥ 1 G (data are based on the service provided by Verizon)

Table 32.2 Survey of wireless technologies in terms of bit rate and range

Types	Bit rate (bps)	Band (Hz)	Range (m)
Zigbee (802.15.4)	20–250 K	868–915 M/2.4 G	50
3G cellphone	400 K–1.15 M	≤ 3.5 G	15 000
Bluetooth	732 K	2.4 G	100
MWBA (802.20)	1 M	≤3.5 G	15 000
WiFi (802.11a,b,g)	11–54 M	2.4 G or 5.8 G	100
WiMax (802.16)	70 M	2–11, 10–66 G	50 000

heterogeneous hosts and OSs including text, image, audio, and video. The protocol has significantly contributed to the boom of the Internet. It also changes the traditional client/server (C/S) communication architecture to a browser/server (B/S) architecture. A typical configuration of the B/S architecture consists of a web server and clients with web browsers. The web server projects the contents in hypertext markup language (HTML) format or its variants, which is transmitted over the Internet using HTTP. User inputs can be acquired using the common gateway interface (CGI) or other variants. The B/S architecture is the most accessible because no specialized software is needed at the client end.

32.3.2 Wired Communication Links

Even during peak usage, the network backbones of the Internet often run at less than 30% of their overall capacity. The average backbone utilization is around 15–20%. The primary speed limitation for the Internet is the *last mile*, the link between clients and their local Internet service providers (ISP).

Table 32.1 lists typical bit rates for different connection types. It is interesting to note the asymmetric speeds in many cases, where upstream bit rate (from the client to the Internet), are far slower than downstream bit rates (from the Internet to the client). These asymmetries introduce complexity into the network model

for teleoperation. Since the speed difference between the slowest modem link and the fastest Internet II node is over 10 000, designers of a networked telerobotic system should anticipate a large variance of communication speeds.

32.3.3 Wireless Links

Table 32.2 compares the speed, band, and range of wireless standards as of 2006. Increasing bit rate and communication range requires increasing power. The amount of radiofrequency (RF) transmission power required over a distance d is proportional to d^k , where $2 \leq k \leq 4$ depending on the antenna type. In Table 32.2, Bluetooth and Zigbee are typical low-power transmission standards that are good for short distances. WiMax and MWBA are currently under development.

By providing high-speed connectivity at low cost, WiFi is the most popular wireless standard in 2006. Its range is approximate 100 m line of sight and the WiFi wireless network usually consists of small-scale interconnected access points. The coverage range usually limits these networks to an office building, home, and other indoor environments. WiFi is a good option for indoor mobile robots and human operators. If the robot needs to navigate in the outdoor environment, the third-generation (3G) cellphone network can provide the best coverage available. Although obvious overlap exists among wireless standards in coverage and band-

width, there are two import issues that have not been covered by Table 32.2. One is mobility. We know that, if an RF source or receiver is moving, the corresponding Doppler effect causes a frequency shift, which could cause problems in communication. WiFi is not designed for fast-moving hosts. WiMax and 3G cellphone allows the host to move at a vehicle speed under 120 km/h. However, MWBA allows the host to move at a speed of 250 km/h, which is the only protocol that works for high-speed trains. Both WiMax and MWBA are designed with a transmission latency of less than 20 ms. However, 3G cellphone networks have a variable latency of 10–500 ms.

32.3.4 Properties of Networked Telerobotics

As defined by Mason, Peshkin, and others [32.43, 44], in *quasistatic* robot systems, accelerations and inertial forces are negligible compared to dissipative forces. In quasistatic robot systems, motions are often modeled as transitions between discrete atomic *configurations*.

We adopt a similar terminology for networked telerobots. In quasistatic telerobotics (QT), robot dynamics

and stability are handled locally. After each atomic motion, a new state report is presented to the remote user, who sends back an atomic command. The atomic state describes the status of the robot and its corresponding environment. Atomic commands refer to human directives, which are desired robotic actions.

Several issues arise

- *State-command presentation*: How should state and available commands be presented to remote human operators using the two-dimensional (2-D) screen display?
- *Command execution/state generation*: How should commands be executed locally to ensure that the desired state is achieved and maintained by the robot?
- *Command coordination*: How should commands be resolved when there are multiple human operators?

32.3.5 Building a Networked Telerobotic System

As illustrated in Fig. 32.4, a typical networked telerobotic system typically includes three components:

- users: anyone with an Internet connection and a web browser
- web server: a computer running a web server software
- robot: a robot manipulator, a mobile robot, or any device that can modify or affect its environment

Users access the system via their web browsers. Any web browser that is compatible with W3C's [HTML](#) standard can access a web server. In 2006, the most popular web browsers are Microsoft Internet Explorer, Netscape, Mozilla Firefox, Safari, and Opera. New browsers and updated versions with new features are introduced periodically.

A web server is a computer that responds to HTTP requests over the Internet. Depending upon the operating system of the web server, popular server software packages include Apache and Microsoft Internet Information Services (IIS). Most servers can be freely downloaded from the Internet.

To develop a networked telerobot, one needs a basic knowledge of developing, configuring, and maintaining web servers. As illustrated in Fig. 32.5, the development requires knowledge of [HTML](#) and at least one local programming languages such as C, [CGI](#), Javascript, Perl, PHP, or Java.

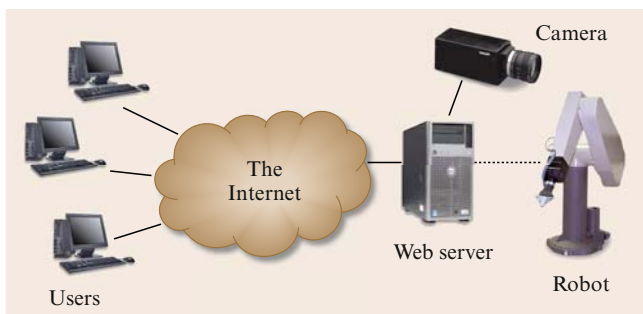


Fig. 32.4 Typical system architecture for a networked telerobot

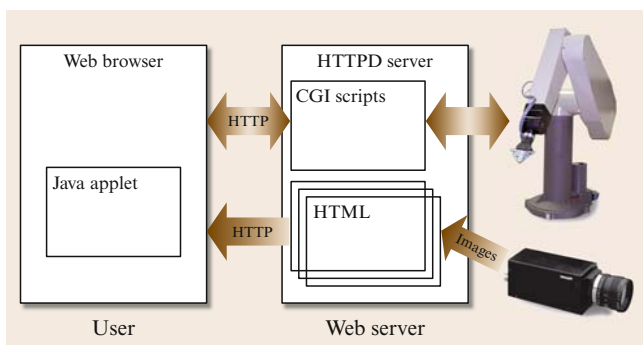


Fig. 32.5 A sample software architecture of a networked telerobot

It is important to consider compatibility with the variety of browsers. Although **HTML** is designed to be compatible with all browsers, there are exceptions. For example, Javascript, which is the embedded scripting language of web browsers, is not completely compatible between Internet Explorer and Netscape. One also needs to master the common **HTML** components such as forms that are used to accept user inputs, frames that are used to divide the interface into different functional regions, etc. An introduction to **HTML** can be found in [32.45].

User commands are usually processed by the web server using **CGI**, the common gateway interface. Most sophisticated methods such as PHP, Java Server Pages (**JSP**), and socket programming can also be used. **CGI** is invoked by the HTTP server when the **CGI** script is referred in the Uniform Resource Locator (**URL**). The **CGI** program then interprets the inputs, which is often the next robot motion command, and sends commands to the robot via a local communication channel. **CGI** scripts can be written in almost any programming language. The most popular ones are Perl and C.

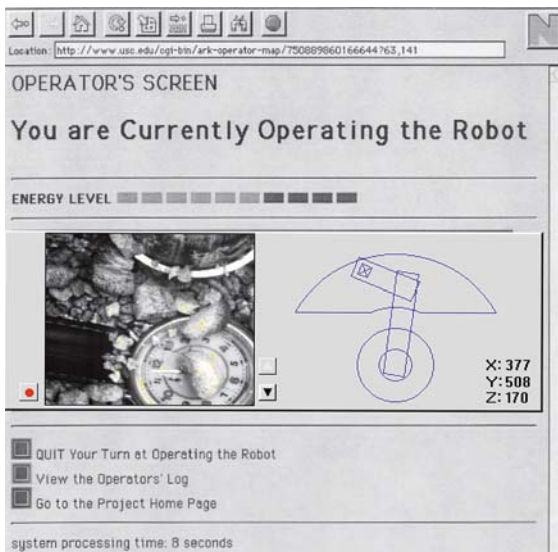


Fig. 32.6 Browser's view of the first networked telerobot interface [32.46]. The schematic at lower right gives an overhead view of position of the four-axis robot arm (with the camera at the end marked with X), and the image at the lower left indicates the current view of the camera. The small button marked with a dot at the left directs a 1 s burst of compressed air into the sand below the camera. The Mercury Project was online from August 1994 to March 1995

A simple networked telerobotic system can be constructed using only **HTML** and **CGI**. However, if the robot requires a sophisticated control interface, a Java applet is recommended. Java applets run inside the web browser on the client's computer. Information about Java can be found at Sun Microsystems' official Java home page.

Most telerobotic systems also collect user data and robot data. Therefore, database design and data processing program are also needed. The most common used databases include MySQL and PostgreSQL. Both are open-source databases and support a variety of platforms and operation systems. Since a networked telerobotic system is online 24 hours a day, reliability is also an important consideration in system design. Website security is critical. Other common auxiliary developments include online documentation, online manual, and user feedback collection.

32.3.6 State-Command Presentation

To generate a correct and high-quality command depends on how effectively the human operator understands the state feedback. The state-command presentation contains three subproblems: the 2-D representation of the true robot state (state display), the assistance provided by the interface to generate new commands (spatial reasoning), and the input mechanism.

Browser Displays

Unlike traditional point-to-point teleoperation, where specialized training and equipment are available to ope-

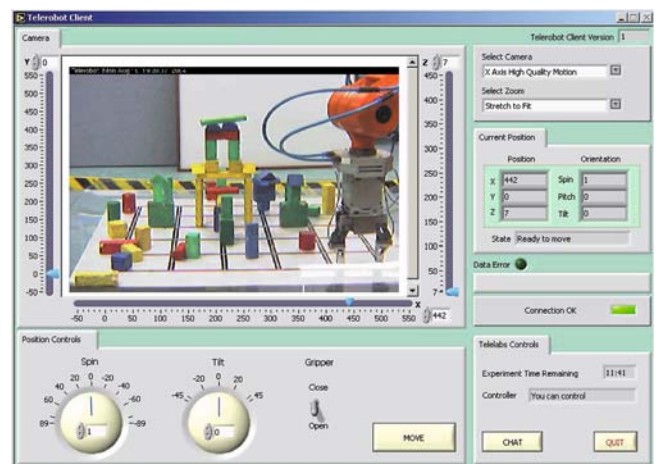


Fig. 32.7 Browser interface to the Australian networked telerobot which was a six-axis arm that could pick up and move blocks [32.16]

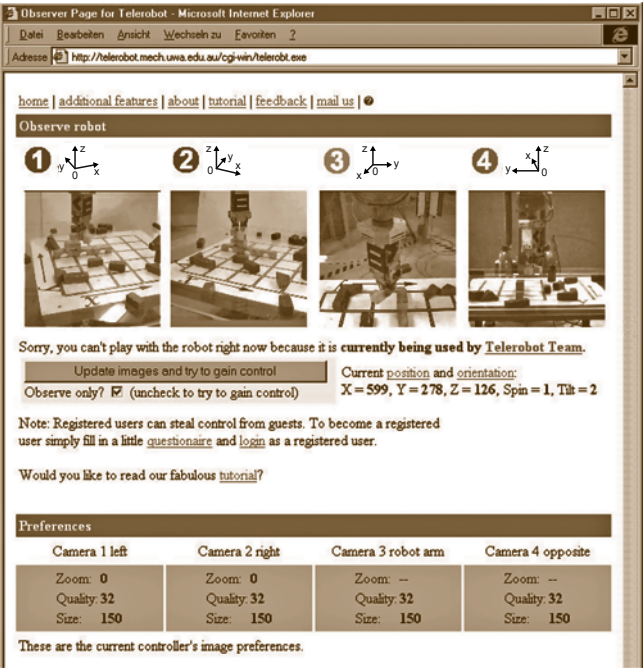


Fig. 32.8 Use of a multicamera system for multi-viewpoint state feedback [32.47]

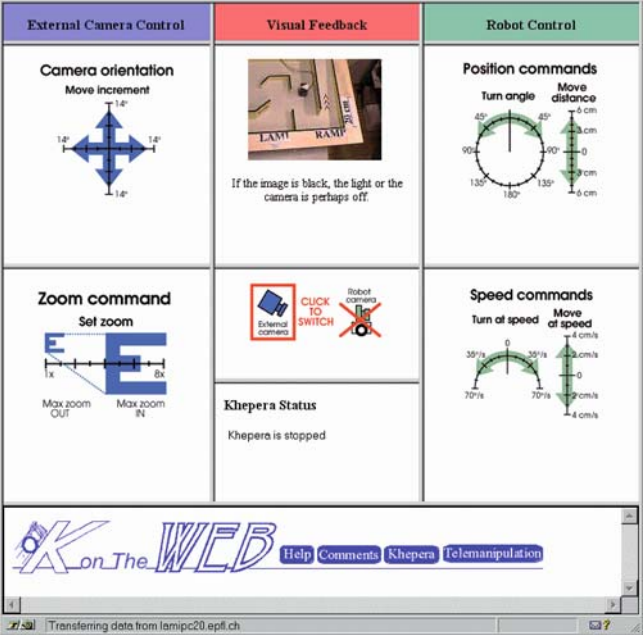


Fig. 32.9 Camera control and mobile robot control in Patrick Saucy and Francesco Mondada's Khep on the web project

rators, networked telerobots offer wide access to the general public. Designers cannot assume that operators have any prior experience with robots. As illustrated in Fig. 32.6, networked telerobotic systems must display the robot state on a 2-D screen display.

The states of the teleoperated robot are often characterized in either world coordinates or robot joint configuration, which are either displayed in numerical format or through a graphical representation. Figure 32.6 lists robot XYZ coordinates on the interface and draws a simple 2-D projection to indicate joint configurations. Figure 32.7 illustrates another example of teleoperation interface that was developed by Taylor and Trevelyan [32.36]. In this interface, XYZ coordinates are presented in a sliding bar near the video window.

The state of the robot is usually displayed in a 2-D view as shown in Figs. 32.6 and 32.7. In some systems, multiple cameras can help the human operator to understand the spatial relationship between the robot and the objects in the surrounding environment. Figure 32.8 shows an example with four distinct camera views for a six-degree-of-freedom industrial robot.

Figure 32.9 demonstrate an interface with a pan-tilt-zoom robotic camera. The interface in Fig. 32.9 is designed for a mobile robot.

More sophisticated spatial reasoning can eliminate the need for humans to provide low-level control by automatically generating a sequence of commands after it receives task-level commands from the human operator. This is particularly important when the robotic system is highly dynamic and requires a very fast response. In this case, it is impossible to ask the human to generate intermediate steps in the robot control; for example, Belousov et al. adopt a shared autonomy model to direct a robot to capture a moving rod [32.27]. Fong and Thorpe [32.48] summarize vehicle teleoperation systems that utilize these supervisory control techniques. Su et al. developed an incremental algorithm for better translation of the intention and motion of operators into remote robot action commands [32.32].

Human Operator Input
Most networked telerobotic systems only rely on mouse and keyboards for input. The design problem is what to click on in the interface. Given the fact that user commands can be quite different, we need to adopt an appropriate interface for inputs; for example, inputs could be Cartesian XYZ coordinates in world coordinate system or robot configurations in angular joint configurations.

For angular inputs, it is often suggested to use a round dial as a control interface, as illustrated in bottom left of Fig. 32.7 and the right-hand side of Fig. 32.9. For linear motion in Cartesian coordinate, arrows operated by either mouse clicks or the keyboard are often suggested. Position and speed control are often needed, as illustrated in Fig. 32.9. Speed control is usually controlled by mouse clicks on a linear progress bar for translation and a dial for rotation.

The most common control type is position control. The most straightforward way is to click on the video image directly. To implement the function, the software needs to translate the 2-D click inputs into three-dimensional (3-D) world coordinates. To simplify the problem, the system designer usually assumes that the clicked position is on a fixed plane; for example, a mouse click on the interface of Fig. 32.6 assumes the robot moves on the X - Y plane. The combination of a mouse click on the image can also allow abstract task-level command. The example in Fig. 32.12 uses mouse clicks to place votes on an image to generate a com-

mand that directs a robot to pick up a test agent at the task level.

32.3.7 Command Execution/State Generation

When a robot receives a command, it executes the command and a new state is generated and transmitted back to the human operator. However, commands may not arrive in time or may get lost in transmission. Also, because users are often inexperienced, their commands may contain errors.

Belousov and colleagues demonstrated a system that allowed a web user to capture a fast rod that is thrown at a robot manipulator [32.27]. Over the limited communication channel, it is impossible to ask the human to control the manipulator directly. Computer vision and augmented-reality-based local intelligence is required

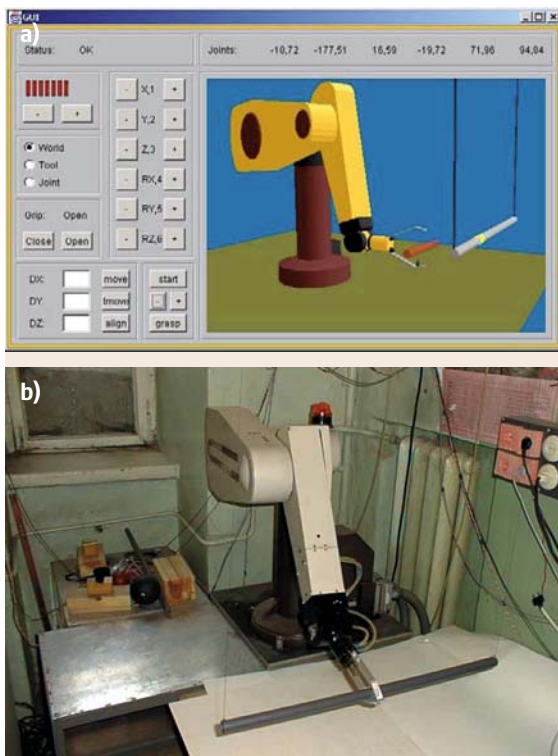


Fig. 32.10a,b A web-based teleoperation system that allows robot to capture a fast-moving rod [32.27] (a) User interface and (b) system setup



Fig. 32.11 Spatial dynamic voting interface for the Tele-Actor system [32.41]: the spatial dynamic voting (SDV) interface as viewed by each user. In the remote environment, the Tele-Actor takes images with a digital camera, which are transmitted over the network and displayed to all participants with a relevant question. With a mouse click, each user places a color-coded marker (a *votel* or voting element) on the image. Users view the position of all votels and can change their votel positions based on the group's response. Votel positions are then processed to identify a *consensus region* in the voting image that is sent back to the Tele-Actor. In this manner, the group collaborates to guide the actions of the Tele-Actor

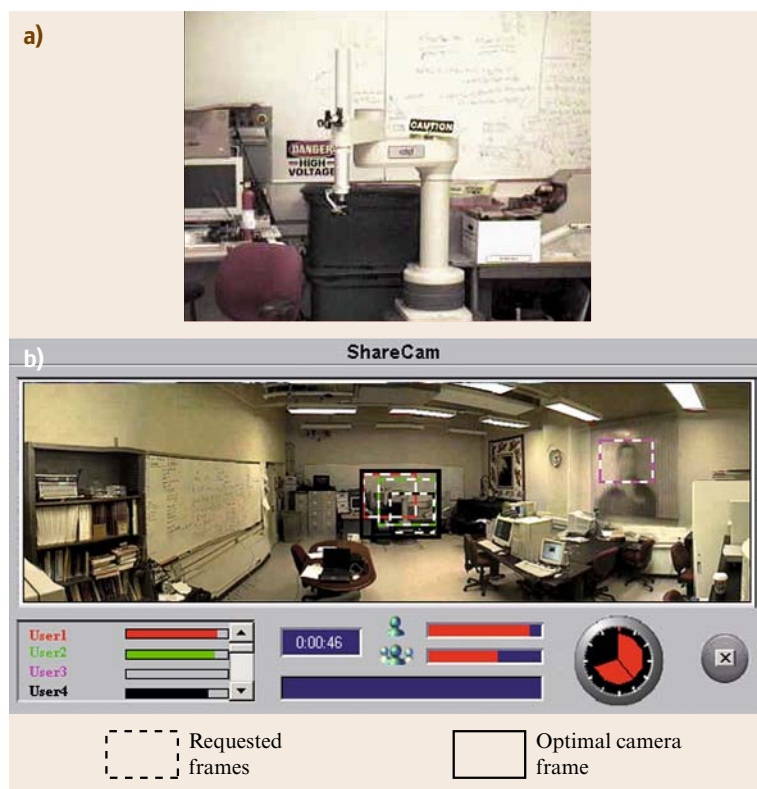


Fig. 32.12a,b Frame selection interface [32.49]. The user interface includes two image windows. The lower window (b) displays a fixed panoramic image based on the camera's full workspace (reachable field of view). Each user requests a camera frame by positioning a *dashed rectangle* in (b). Based on these requests, the algorithm computes an optimal camera frame (shown with a *solid rectangle*), moves the camera accordingly, and displays the resulting live streaming video image in the upper window (a)

to assist the human operator. The rod is on bifilar suspension, performing complicated oscillations. Belousov et al. designed a shared-autonomy control to implement the capture. First, an operator chooses the desired point for capture on the rod and the capture instant using a 3-D online virtual model of the robot and the rod. Then, the capturing operation is performed automatically using a motion prediction algorithm that is based on the rod's motion model and two orthogonal camera inputs, which perceive the rod's position locally in real time.

32.3.8 Collaborative Control

When more than one human is sharing control of the device, command coordination is needed. According to [32.50], multiple human operators can reduce the chance of errors, cope with malicious inputs, utilize operators' different expertise, and train new operators. In [32.51], a *collaborative telerobot* is defined as a tele-robot simultaneously controlled by many participants, where input from each participant is combined to generate a single control stream.

When group inputs are in the form of direction vectors, averaging can be used as an aggregation mechanism [32.52]. When decisions are distinct choices or at the abstract task level, voting is a better choice [32.41]. As illustrated in Fig. 32.12, Goldberg and Song develop the Tele-Actor system using spatial dynamic voting. The Tele-Actor is a human equipped with an audio/video device and controlled by a group of online users. Users indicate their intentions by positioning their votes on a 320×320 pixel voting image during the voting interval. Votes are collected at the server and used to determine the Tele-Actor's next action based on the most requested region on the voting image. (see <http://www.tele-actor.net>)

Song and Goldberg [32.49, 53] developed a controllable camera that allowed many clients to share control of its camera parameters, as illustrated in Fig. 32.12. Users indicate the area they want to view by drawing rectangles on a panoramic image. The algorithm computes an optimal camera frame with respect to the user satisfaction function, which is defined as the frame selection problem.

32.4 Conclusion and Future Directions

As this technology matures, networked telerobots will gradually go beyond university laboratories and find application in the real world. A new project, the Collaborative Observatory for Nature Environments (CONE) project, proposed by Song and Goldberg [32.54], aims to design a networked robotic camera system to collect data from the wilderness for natural scientists. The fast development of networked telerobot system is not limited to North America. Japan's Advanced Telecommunications Research Institute International (ATR) Intelligent Robotics and Communication Laboratory has also announced its networked robot project led by Norihiro Hagita (ATR). Its mission is to develop network-based intelligent robots for applications such as service, medical, and safety. Hideyuki Tokuda (Keio University) chaired the Networked Robot Forum in Spring 2005, which promotes research and development (R&D) and standardization on network robots through activities to support awareness campaigns and verification experiments in collaboration among wide-ranging parties, which includes over 100 industry and academic members. Korea's Ministry of Information and Communication has also announced the Ubiquitous Robotic Companion (URC) project to develop network-based intelligent robots.

Networked telerobots have allowed tens of thousands of nonspecialists around the world to interact with robots. The design of networked telerobots presents a number of engineering challenges to build reliable systems that can be operated by nonspecialists 24 hours a day, 7 days a week and remain online for years. Many new research challenges remain.

- *New interfaces:* As portable devices such as cell-phones and portable digital assistants (PDAs) becomes grow in computation power, networked telerobotics should be able to adopt them as new

interfaces. As computers becomes increasingly powerful, they become capable of visualizing more sophisticated sensor inputs. Designers of new interfaces should also keep track of new developments in hardware such as haptic interfaces and voice recognition systems. New software standards such as flash, extensible markup language (XML), extensible hyper text markup language (XHTML), virtual reality modeling language (VRML), and wireless markup language (WML) will also change the way we design interface.

- *New algorithms:* Algorithms determine performance. Scalable algorithms that are capable of handling large amounts of data such as video/sensor network inputs and utilize fast-evolving hardware capability such as distributed and parallel computation will become increasingly important in the networked telerobotics.
- *New protocols:* Although we have listed some pioneering work in changing the network environment to improve teleoperation, there are still a large number of open problems such as new protocols, appropriate bandwidth allocation [32.55], QoS [32.56], security, routing mechanisms [32.28], and many more. Network communication is a very fast-evolving field. The incorporation/modification of network communication ideas into networked telerobotic system design will continue to be an active research area. The common object request broker architecture (CORBA) or real-time CORBA [32.19, 20, 57, 58] have great potential for networked telerobotics.
- *Applications:* Many new applications are emerging in areas such as security, inspection, education, and entertainment. Application requirements such as reliability, security, and modularity will continuous to pose new challenges for system design.

References

- | | |
|---|---|
| <p>32.1 K. Goldberg, R. Siegwart (Eds.): <i>Beyond Webcams: An Introduction to Online Robots</i> (MIT Press, Cambridge 2002)</p> <p>32.2 N. Tesla: Method of an apparatus for controlling mechanism of moving vessels or vehicles http://www.pbs.org/tesla/res/613809.html (1898)</p> <p>32.3 R. Goertz, R. Thompson: Electronically controlled manipulator, <i>Nucleonics</i> 12(11), 46–47 (1954)</p> | <p>32.4 R.D. Ballard: A last long look at titanic, <i>National Geographic</i> 170(6), 698–727 (1986)</p> <p>32.5 A.K. Bejczy: Sensors, controls, and man-machine interface for advanced teleoperation, <i>Science</i> 208(4450), 1327–1335 (1980)</p> <p>32.6 R.S. Moshier: Industrial manipulators, <i>Sci. Am.</i> 211(4), 88–96 (1964)</p> <p>32.7 R. Tomovic: On man-machine control, <i>Automatica</i> 5(4), 40–404 (1969)</p> |
|---|---|

- 32.8 A. Bejczy, G. Bekey, R. Taylor, S. Rovetta: A research methodology for tele-surgery with time delays, First Int. Symp. Med. Robot. Comp. Assist. Surg. (1994)
- 32.9 M. Gertz, D. Stewart, P. Khosla: A human-machine interface for distributed virtual laboratories, IEEE Robot. Autom. Mag. **1**(4), 5–13 (1994)
- 32.10 T. Sato, J. Ichikawa, M. Mitsuishi, Y. Hatamura: A new micro-teleoperation system employing a hand-held force feedback pencil, IEEE Int. Conf. Robot. Autom. (1994)
- 32.11 T.B. Sheridan: *Telerobotics, Automation, and Human Supervisory Control* (MIT Press, Cambridge 1992)
- 32.12 <http://www.cl.cam.ac.uk/coffee/qsf/timeline.html>
- 32.13 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Robot teleoperation via WWW, IEEE Int. Conf. Robot. Autom. (1995)
- 32.14 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Beyond the web: Manipulating the physical world via the WWW, Comp. Netw. ISDN Syst. J. **28**(1), 209–219 (1995), Archives can be viewed at <http://www.usc.edu/dept/raiders/>
- 32.15 B. Dalton, K. Taylor: A framework for internet robotics, IEEE Int. Conf. Intell. Robot. Syst. (IROS): Workshop on Web Robots (Victoria 1998)
- 32.16 <http://telerobot.mech.uwa.edu.au/>
- 32.17 H. Hu, L. Yu, P.W. Tsui, Q. Zhou: Internet-based robotic systems for teleoperation, Assembly Autom. **21**(2), 143–151 (2001)
- 32.18 R. Safaric, M. Debevc, R. Parkin, S. Uran: Telerobotics experiments via internet, IEEE Trans. Ind. Electron. **48**(2), 424–431 (2001)
- 32.19 S. Jia, K. Takase: A CORBA-based internet robotic system, Adv. Robot. **15**(6), 663–673 (2001)
- 32.20 S. Jia, Y. Hada, G. Ye, K. Takase: Distributed telecare robotic systems using CORBA as a communication architecture, IEEE Int. Conf. Robot. Autom. (ICRA) (Washington 2002)
- 32.21 J. Kim, B. Choi, S. Park, K. Kim, S. Ko: Remote control system using real-time mpeg-4 streaming technology for mobile robot, IEEE Int. Conf. Consum. Electron. (2002)
- 32.22 T. Mirfakhrai, S. Payandeh: A delay prediction approach for teleoperation over the internet, IEEE Int. Conf. Robot. Autom. (ICRA) (2002)
- 32.23 K. Han, Y. Kim, J. Kim, S. Hsia: Internet control of personal robot between kaist and uc davis, IEEE Int. Conf. Robot. Autom. (ICRA) (2002)
- 32.24 L. Ngai, W.S. Newman, V. Liberatore: An experiment in internet-based, human-assisted robotics, IEEE Int. Conf. Robot. Autom. (ICRA) (2002)
- 32.25 R.C. Luo, T.M. Chen: Development of a multibehavior-based mobile robot for remote supervisory control through the internet, IEEE/ASME Trans. Mechatron. **5**(4), 376–385 (2000)
- 32.26 D. Aarno, S. Ekvall, D. Kragi: Adaptive virtual fixtures for machine-assisted teleoperation tasks, IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 1151–1156
- 32.27 I. Belousov, S. Chebukov, V. Sazonov: Web-based teleoperation of the robot interacting with fast moving objects, IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 685–690
- 32.28 Z. Cen, A. Goradia, M. Mutka, N. Xi, W. Fung, Y. Liu: Improving the operation efficiency of supermedia enhanced internet based teleoperation via an overlay network, IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 691–696
- 32.29 N.P. Jouppi, S. Thomas: Telepresence systems with automatic preservation of user head height, local rotation, and remote translation, IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 62–68
- 32.30 B. Ricks, C.W. Nielsen, M.A. Goodrich: Ecological displays for robot interaction: a new perspective, Int. Conf. Intell. Robot. Syst. (IROS), Vol. 3 (2004) pp. 2855–2860
- 32.31 D. Ryu, S. Kang, M. Kim, J. Song: Multi-modal user interface for teleoperation of robhaz-dt2 field robot system, Int. Conf. Intell. Robot. Syst. (IROS), Vol. 1 (2004) pp. 168–173
- 32.32 J. Su, Z. Luo: Incremental motion compression for telepresent walking subject to spatial constraints, IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 69–74
- 32.33 I. Toshima, S. Aoki: Effect of driving delay with an acoustical tele-presence robot, telehead, IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 56–61
- 32.34 N. Chong, T. Kotoku, K. Ohba, K. Komoriya, N. Matsuhira, K. Tanie: Remote coordinated controls in multiple telerobot cooperation, IEEE Int. Conf. Robot. Autom., Vol. 4 (2000) pp. 3138–3343
- 32.35 E. Paulos, J. Canny, F. Barrientos: Prop: Personal roving presence, SIGGRAPH Vis. Proc. (1997) p. 99
- 32.36 K. Taylor, J.P. Trevelyan: Australia's telerobot on the web, 26th Symp. Ind. Robot. (Singapore 1995) pp. 39–44
- 32.37 A. Khamis, D.M. Rivero, F. Rodriguez, M. Salichs: Pattern-based architecture for building mobile robotics remote laboratories, IEEE Int. Conf. Robot. Autom. (ICRA) (Taipei 2003) pp. 3284–3289
- 32.38 C. Cosma, M. Confente, D. Botturi, P. Fiorini: Laboratory tools for robotics and automation education, IEEE Int. Conf. Robot. Autom. (ICRA) (Taipei 2003) pp. 3303–3308
- 32.39 K.W. Dorman, J.L. Pullen, W.O. Keks, P.H. Eismann, K.A. Kowalski, J.P. Karlen: The servicing aid tool: A teleoperated robotics system for space applications, The Seventh Annual Workshop on Space Operations Applications and Research (SOAR 1993), Vol. 1 (Johnson Space Center, Houston 1994)
- 32.40 C. Pollak, H. Hutter: A webcam as recording device for light microscopes, J. Comp. Assist. Microsc. **10**(4), 179–183 (1998)

- 32.41 K. Goldberg, D. Song, A. Levandowski: Collaborative teleoperation using networked spatial dynamic voting, *Proc. IEEE* **91**(3), 430–439 (2003)
- 32.42 J. Walrand, P. Varaiya: *High-Performance Communication Networks*, 2nd edn. (Morgan Kaufmann, San Francisco 2000)
- 32.43 M.A. Peshkin, A.C. Sanderson: Minimization of energy in quasi-static manipulation, *IEEE Trans. Robot. Autom.* **5**(1), 53–60 (1989)
- 32.44 M.T. Mason: On the scope of quasi-static pushing, 3rd Int. Symp. Robot. Res., ed. by O. Faugeras, G. Giralt (MIT Press, Cambridge 1986)
- 32.45 E. Ladd, J. O'Donnell: *Using HTML 4, XML, and Java 1.2* (QUE Press, 1998)
- 32.46 K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, J. Wiegley: Desktop tele-operation via the world wide web, *IEEE Int. Conf. Robot. Autom.* (Nagoya 1995)
- 32.47 H. Friz: Design of an Augmented Reality User Interface for an Internet Based Telerobot Using Multiple Monoscopic Views. Ph.D. Thesis (Technical University of Clausthal, Clausthal-Zellerfeld 2000)
- 32.48 T. Fong, C. Thorpe: Vehicle teleoperation interfaces, *Auton. Robot.* **11**, 9–18 (2001)
- 32.49 D. Song, A. Pashkevich, K. Goldberg: Sharecam part II: Approximate and distributed algorithms for a collaboratively controlled robotic webcam, *IEEE/RSJ Int. Conf. Intell. Robot. (IROS)*, Vol.2 (Las Vegas 2003) pp.1087–1093
- 32.50 K. Goldberg, B. Chen, R. Solomon, S. Bui, B. Farzin, J. Heitler, D. Poon, G. Smith: Collaborative teleoperation via the internet, *IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol.2 (2000) pp.2019–2024
- 32.51 D. Song: Systems and Algorithms for Collaborative Teleoperation. Ph.D. Thesis (Department of Industrial Engineering and Operations Research, University of California 2004)
- 32.52 K. Goldberg, B. Chen: Collaborative teleoperation via the internet, *Int. Conf. Intell. Robot. Syst. (IROS)* (2001)
- 32.53 D. Song, K. Goldberg: Sharecam part I: Interface, system architecture, and implementation of a collaboratively controlled robotic webcam, *IEEE/RSJ Int. Conf. Intell. Robot. (IROS)*, Vol.2 (Las Vegas 2003) pp.1080–1086
- 32.54 D. Song, K. Goldberg: CONE Project (www.c-o-n-e.org)
- 32.55 P.X. Liu, M. Meng, S.X. Yang: Data communications for internet robots, *Auton. Robot.* **15**, 213–223 (2003)
- 32.56 W. Fung, N. Xi, W. Lo, B. Song, Y. Sun, Y. Liu, I.H. Elhajj: Task driven dynamic qos based bandwidth allocation for real-time teleoperation via the internet, *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Las Vegas 2003)*
- 32.57 M. Amoretti, S. Bottazzi, M. Reggiani, S. Caselli: Evaluation of data distribution techniques in a CORBA-based telerobotic system, *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Las Vegas 2003)*
- 32.58 S. Bottazzi, S. Caselli, M. Reggiani, M. Amoretti: A software framework based on real time COBRA for telerobotics systems, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, EPFL (Lausanne 2002)

Exoskeletons

33. Exoskeletons for Human Performance Augmentation

Homayoon Kazerooni

Although autonomous robotic systems perform remarkably in structured environments (e.g., factories), integrated human–robotic systems are superior to any autonomous robotic systems in unstructured environments that demand significant adaptation. The technology associated with exoskeleton systems and human power augmentation can be divided into lower–extremity exoskeletons and upper–extremity exoskeletons. The reason for this was twofold; firstly, one could envision a great many applications for either a stand-alone lower- or upper-extremity exoskeleton in the immediate future. Secondly, and more importantly for the division, is that these exoskeletons are in their early stages, and further research still needs to be conducted to ensure that the upper-extremity exoskeleton and lower-extremity exoskeleton can function well independently before one can venture an attempt to integrate them. This chapter first gives a description of the upper-extremity exoskeleton efforts and then will proceed with the

- 33.1 Survey of Exoskeleton Systems 773
- 33.2 Upper-Extremity Exoskeleton 775
- 33.3 Intelligent Assist Device 776
- 33.4 Control Architecture for Upper-Extremity Exoskeleton Augmentation..... 778
- 33.5 Applications of Intelligent Assist Device .. 780
- 33.6 Lower-Extremity Exoskeleton 780
- 33.7 The Control Scheme of an Exoskeleton.... 782
- 33.8 Highlights of the Lower-Extremity Design 786
- 33.9 Field-Ready Exoskeleton Systems..... 790
 - 33.9.1 The ExoHiker Exoskeleton 790
 - 33.9.2 The ExoClimber Exoskeleton..... 790
- 33.10 Conclusion and Further Reading 792
- References 792

more detailed description of the lower-extremity exoskeleton.

33.1 Survey of Exoskeleton Systems

In the early 1960s, the US Defense Department expressed interest in the development of a man-amplifier, a *powered suit of armor* which would augment soldiers’ lifting and carrying capabilities. In 1962, the Air Force had the Cornell Aeronautical Laboratory study the feasibility of using a master–slave robotic system as a man-amplifier. In later work, Cornell determined that an exoskeleton, an external structure in the shape of the human body which has far fewer degrees of freedom than a human, could accomplish most desired tasks [33.1]. From 1960 to 1971, General Electric developed and tested a prototype man-amplifier, a master–slave system called the Hardiman [33.2–6]. The Hardiman was a set of overlapping exoskeletons worn by a human operator.

The outer exoskeleton (the slave) followed the motions of the inner exoskeleton (the master), which followed the motions of the human operator. All these studies found that duplicating all human motions and using master–slave systems were not practical. Additionally, difficulties in human sensing and system complexity kept it from walking.

Vukobratovic et al. developed a few active orthoses for paraplegics [33.7]. The systems include hydraulic or pneumatic actuators for driving the hip and knee joints in the sagittal plane. These orthoses were coupled with the wearer via shoe bindings, cuffs, and a corset. The device was externally powered and controlled via a predetermined periodic motion. Although these early devices

were limited to predefined motions and had limited success, balancing algorithms developed for them are still used in many bipedal robots [33.8].

Seireg et al. also created an exoskeleton system for paraplegics where only the hip and knee were powered by hydraulic actuators in sagittal plane [33.9]. The hydraulic power unit consists of a battery-powered direct-current (DC) motor, pump, and accumulator. A bank of servo-valves drives the actuators at the knee and hip. The device was controlled to follow a set of

joint trajectories without the use of any sensory systems from its wearer.

The hybrid assisted limb (HAL) was developed at the University of Tsukuba ([33.13, 14]). This 15 kg battery-powered suit detects muscle myoelectrical signals on the skin surface below the hip and above the knee. The signals are picked up by the sensors and sent to the computer, which translates the nerve signals into signals of its own for controlling electric motors at the hips and knees of the exoskeleton, effectively amplify-

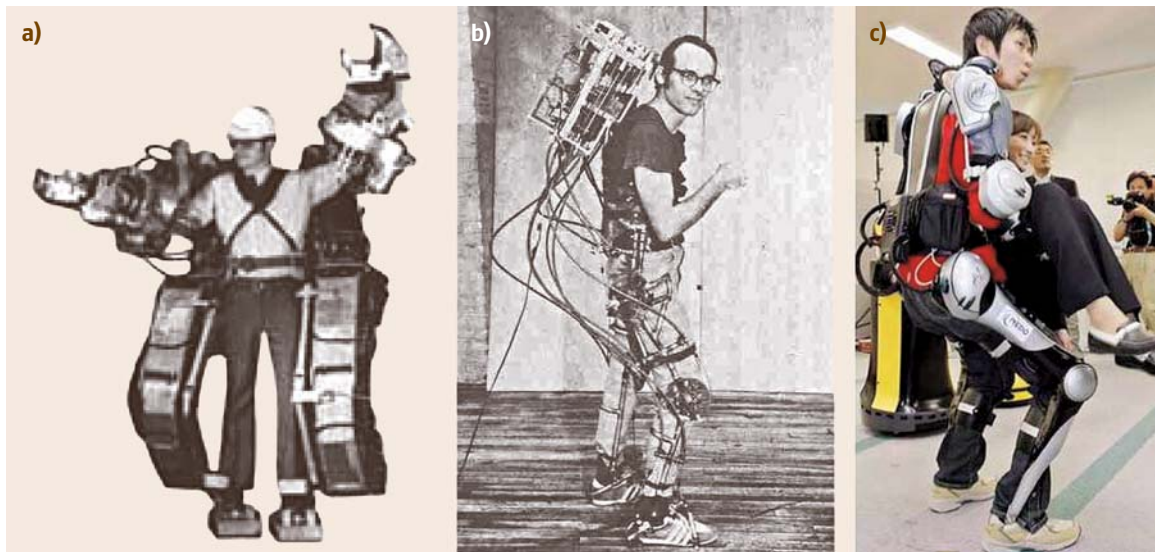


Fig. 33.1 (a) Hardiman; (b) An exoskeleton system designed for paraplegics by *Seireg* et al. [33.9]; (c) HAL

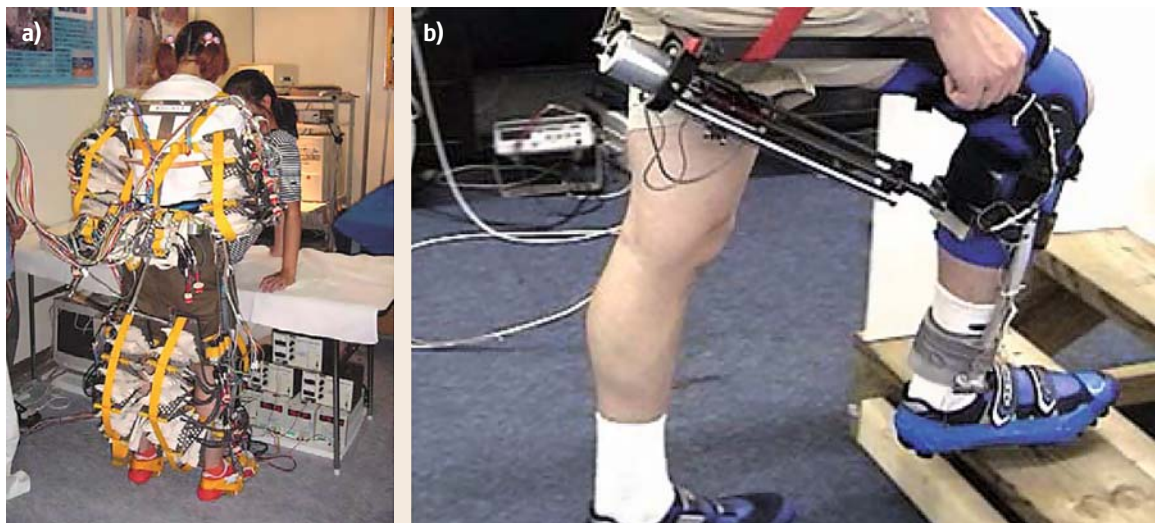


Fig. 33.2 (a) An exoskeleton for patient handling [33.10, 11]; (b) RoboKnee [33.12]

ing muscle strength. In addition to electromyography (EMG) signals, the device further includes potentiometers for measuring the joint angles, force sensors for measuring the ground reaction forces and a gyroscope and accelerometer for measuring the torso angle. Each leg of HAL powers the flexion/extension motion at the hip and knee in the sagittal plane through the use of DC motors integrated with harmonic drives. The ankle includes passive degrees of freedom.

Yamamoto et al. [33.10, 11] have created an exoskeleton system for assisting nurses during patient handling. The lower limbs include pneumatic actuators for the flexion/extension of the hips and knees in the sagittal plane. Air pumps are mounted directly onto each actuator to provide pneumatic power. User input is determined via force sensing resistors coupled to the wearer's skin. The measurement from force sensing resistor (FSR) and other information such as joint angles are used to determine the required input torques for various joints.

Pratt et al. developed a powered knee brace for adding power at the knee to assist in squatting [33.12]. The device is powered by a linear series-elastic actua-

tor coupling the upper and the lower portions of a knee brace. The control of this powered knee brace requires the ground reaction force measured by two load cells. The system uses a positive-feedback force controller to create an appropriate force for the actuator.

Kong et al. developed a full lower-limb exoskeleton system that works with a powered walker [33.15]. The walker houses the electric actuators, the controller, and the batteries, reducing the weight of the exoskeleton system. A transmission system transmits power to the wearer's joints from the actuators in the walker. The exoskeleton is powered at the hips and knees in sagittal plane. The input to drive the system is a set of pressure sensor that measure the force applied by the quadriceps muscle on the knee.

Agrawal et al. have conducted research projects on statically balanced leg orthoses that allow for less effort during swing [33.16]. In the passive version, the device uses springs in order to cancel the gravity force associated with the device links and the person leg. Through experiments the authors showed that the device reduced the required torque by the wearer substantially.

33.2 Upper-Extremity Exoskeleton

In the mid-1980s, researchers at Berkeley initiated several research projects on upper-extremity exoskeleton systems, billed as *human extenders* [33.17–23]. The main function of an upper-extremity exoskeleton is human power augmentation for the manipulation of heavy and bulky objects. Since upper-extremity exoskeletons are mostly used for factory floors, warehouse, and distribution centers, they are hung from overhead cranes. As can be seen in later sections, lower-extremity exoskeletons focus on supporting and carrying heavy payloads on the operator's back (like a backpack) during long-distance locomotion. Upper-extremity exoskeletons, which are also known as assist devices or human power extenders, can simulate forces on a worker's arms and torso. These forces differ from, and are usually much smaller than the forces needed to maneuver a load. When a worker uses an upper-extremity exoskeleton to move a load, the device bears the bulk of the weight by itself, while transferring to the user as a natural feedback a scaled-down value of the load's actual weight. For example, for every 20 kg of weight from an object, a worker might support only 2 kg while the device supports the remaining 18 kg. In this fashion, the worker can still sense the load's weight and judge his/her

movements accordingly, but the force he/she feels is much smaller than what he/she would feel without the device. In another example, suppose the worker uses the device to maneuver a large, rigid, and bulky object, such as an exhaust pipe. The device will convey the force to the worker as if it was a light, single-point mass. This limits the cross-coupled and centrifugal forces that increase the difficulty of maneuvering a rigid body and can sometimes produce injurious forces on the wrist. In a third example, suppose a worker uses the device to handle a powered torque wrench. The device will decrease and filter the forces transferred from the wrench to the worker's arm so the worker feels the low-frequency components of the wrench's vibratory forces instead of the high-frequency components that produce fatigue [33.24]. These assist devices not only filter out unwanted forces on a worker, but can also be programmed to follow a particular trajectory regardless of the exact direction in which the worker attempts to manipulate the device. For example, suppose an auto-assembly worker is using an assist device to move a seat to its final destination inside a car. The assist device can bring the seat to its final destination, moving it along a preprogrammed path with a speed that is proportional



Fig. 33.3a,b Two-handed upper-extremity exoskeleton where artificially built friction forces between the load and the arms allow for grasping objects [33.25]

to the magnitude of the worker's force on the device. Although the worker might be paying very little attention to the final destination of the seat, the device can still bring the seat to its proper place without the worker's guidance. The upper-extremity exoskeleton reflects on the worker's arm forces that are limited and much smaller than the forces needed to maneuver loads. With it, auto-assembly and warehouse workers can maneuver parts and boxes with greatly improved dexterity and precision, not to mention a marked decrease in muscle strain.

33.3 Intelligent Assist Device

The intelligent assist devices (IAD) are the simplest non-anthropomorphic form of the upper-extremity systems that augments human capabilities [33.30, 31]. Figure 33.5 illustrates an intelligent assist device (IAD). At the top of the device, a computer-controlled electric actuator is attached directly to a ceiling, wall, or an overhead crane and moves a strong wire rope precisely, and with a controllable speed. Attached to the wire rope is a sensory end-effector where the operator hand, the IAD, and the load come into contact. The end-effector includes a load interface subsystem and an operator interface subsystem. The load interface subsystem is designed to interface with a variety of loads and holding devices. Hooks, suction cups, and grippers are examples of other connections to the end-effector as shown in Fig. 33.6. In general, to grab complex objects, special tooling systems should be made and connected to the load interface subsystem. The operator interface sub-



Fig. 33.4 One-handed upper-extremity exoskeleton where a gripper allows for grasping of heavy objects [33.21]

The upper-extremity exoskeleton will significantly reduce the incidence of back injury in the workplace, which will in turn greatly decrease the annual cost of treating back injuries.

Upper-extremity exoskeletons were designed based primarily on compliance control [33.26–29] schemes that relied on the measurement of interaction force between the human and the machine. Various experimental systems, including a hydraulic loader designed for loading aircrafts and an electric power extender built for two-handed operation, were designed to verify the theories (Fig. 33.3 and Fig. 33.4).

system includes an ergonomic handle, which contains a high-performance sensor for measuring the magnitude of the vertical force exerted on the handle by the operator. A signal representing the operator force is transmitted to a computer controller, which controls the actuator of the IAD. Using the measurement of the operator force and other calculations, the controller assigns the necessary speed to either raise or lower the wire rope to create enough mechanical strength to assist the operator in the lifting task as required. If the operator pushes upwardly on the handle, the assist device lifts the load; and if the operator pushes downward on the handle, the assist device lowers the load. The load moves appropriately so that only a small preprogrammed proportion of the load force (weight plus acceleration) is supported by the operator, and the remaining force is provided by the actuator of the IAD. All of this happens so quickly that the operator's lifting efforts and the device's lifting efforts are

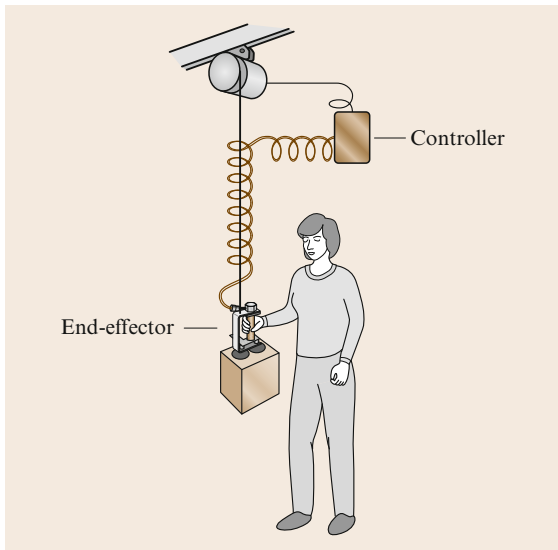


Fig. 33.5 Intelligent assist device: the simplest form of upper-extremity enhancers for industrial applications. The IAD can follow a worker's high-speed maneuvers very closely during manipulations without impeding the worker's motion.

synchronized perfectly and the load feels substantially lighter to the operator. With this load-sharing concept, the operator has the sense that he or she is lifting the load, but with far less force than would ordinarily be required. For example, with a 25 kg load force (gravity plus acceleration), the IAD supports 24 kg, while the operator supports and feels only 1 kg. With the assistance of the

intelligent assist device, a worker can manipulate any object in the same natural way that he/she would manipulate a lightweight object without any assistance. There are no push buttons, keyboards, switches, or valves to control the motion of the intelligent assist device; the user's natural movements, in conjunction with the device computer, controls the motion of the device and its load.

Figure 33.6 shows the end-effector that measures the operator forces at all times even in the presence of loading and unloading shock forces. This robust end-effector also includes a dead-man switch, which is installed on the handle and sends a signal to the controller via a signal cable. If the dead-man switch on the end-effector is not depressed, (i.e., if the operator is not holding onto the handle of the end-effector), the device will be suspended without any motion even if loads are added to or removed from the end-effector.

The IAD is engineered with variety of embedded safety features. One of the most important safety characteristics of the IAD is that the wire rope does not become slack if the end-effector is physically constrained from moving downward and the end-effector is pushed downward by the operator. Slack in the wire rope can have far more serious consequences than slowing down the workers at their jobs; the slack line could wrap around the operator's neck or hand, creating serious or even deadly injuries. The control algorithm in the computer of the IAD, employing the information from various sensors, ensures that the wire rope will never become slack [33.32].

Another form of IAD can be seen in Fig. 33.7 where a sensory glove measures the force the wearer imposes

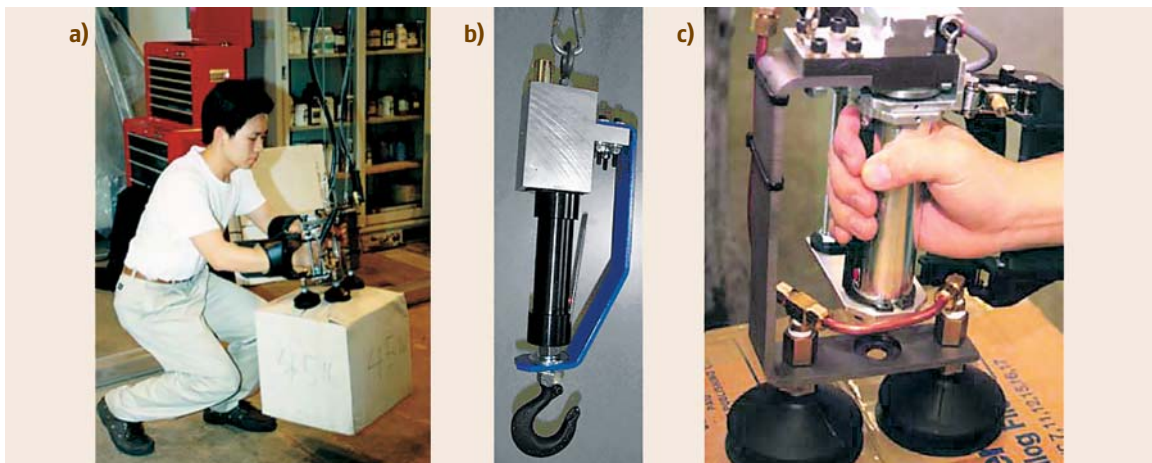


Fig. 33.6a–c The end-effector (a) contains a sensor (b) that measures the force that the operator applies to the handle (c) in the vertical direction

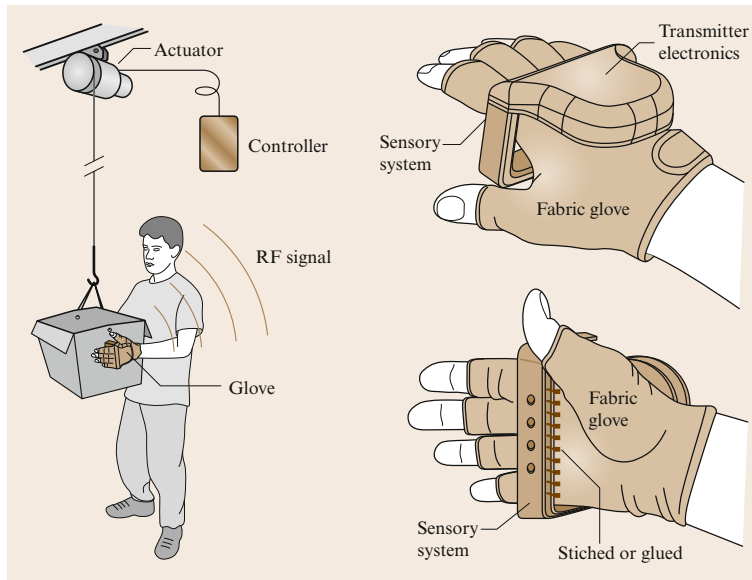


Fig. 33.7 An instrumented glove allows an operator to lift and lower objects naturally while using a hoist, similar to the way one maneuvers objects manually without activating switches or push buttons [33.34]

on any part of the material handling system or the object being maneuvered [33.33, 34]. This instrumented glove is always worn by the operator and therefore remains with the operator. The instrumented glove generates a set of signals as a function of the contact force between the glove and the object being manipulated or the material handling device itself. A set of signals representing the contact force is transmitted in the form of radiofrequency (RF) signals to a device controller so that a command signal is generated. The command sig-

nal is sent to the device actuator to provide the required assistance to maneuver or lift the load as a function of the force imposed by the operator, so that the operator provides only a small portion of the total force needed to maneuver the device and the object being manipulated by the device. For a person observing the operator and the IAD, this interaction seems rather magical since the device responds to the operator's touch regardless of whether the operator is pushing on the IAD or on the object being lifted by the device.

33.4 Control Architecture for Upper-Extremity Exoskeleton Augmentation

The linear system theory is employed here to model the dynamic behavior of the elements of an IAD. This allows us to disclose the system properties in their simplest and most commonly used form. The more general approach (nonlinear and multivariable models for upper-extremity assist devices) are presented in [33.19, 20], and [33.21], where they have been applied to the devices shown in Figs. 33.3 and 33.4. The block diagram of Fig. 33.8 shows the basic control technique. As discussed earlier, the force-sensing element in the end-effector delivers a signal to the controller, which is used to control the actuator. If e is the input command to the actuator, then the linear velocity of the end-effector v can be represented

by:

$$v = Ge + Sf_R, \quad (33.1)$$

where G is the actuator transfer function relating the input command to the actuator to the end-effector velocity; S is the actuator sensitivity transfer function relating the wire rope tensile force f_R to the end-effector velocity, v . A positive value for v represents a downward speed for the load. Also note that, since the load is connected to the end-effector, both terminologies *load velocity* and *end-effector velocity* refer to v as derived by (33.1). If a closed-loop velocity controller is designed for the actuator such that S is small, the actuator has only a small response to the line tensile

upward object speed for a given operator force on the object. As Fig. 33.8 indicates, K may not be arbitrarily large. Rather, the choice of K must guarantee the closed-loop stability of the system. The human force f is a function of the human arm impedance H , whereas

the load force is a function of load dynamics, i. e., the weight and inertial forces generated by the load. One can find many methods to design the controller transfer function K . Reference [33.19] describes the conditions for the closed-loop stability of such systems.

33.5 Applications of Intelligent Assist Device

The IAD was designed with one vision in mind: minimizing the risk of injuries associated with repeated maneuvers, and maximizing the throughput while maintaining robustness and user-friendliness. The IAD has been evaluated extensively for three applications: warehousing and distribution centers, auto-assembly plants, and delivery services. A study on warehousing maneuvers at distribution centers demonstrated that palletizing, depalletizing, loading and unloading trucks, and placing boxes on and off of conveyor belts are the most common maneuvers. Initial studies of the distribution centers demonstrated that objects to be maneuvered in warehouses and distribution centers are mostly boxes weighing less than 27 kg that require workers to maneuver them rapidly (sometimes up to 15 boxes a minute). The use of the IADs in warehouses would have a considerable impact on reducing injuries to the worker population because of the large number of warehouse workers. Figure 33.6 shows the use of the IAD in a distribution center during a depalletizing operation.

Studies of auto-assembly maneuvers revealed that the installation of batteries, gas tanks, bumpers, instrument panels, exhaust pipes, and prop shafts are important maneuvers that would benefit from IADs (Fig. 33.9). Various load interface subsystems must be employed for connection to various auto parts.

Postal services across the world use sacks and trays to hold letters, magazines, and small boxes. These sacks and trays, which are manually handled by mail handlers, are usually fully filled with magazine bundles, envelopes and parcels, and can weigh up to 32 kg. In general, at all distribution centers, several factors contribute to the cre-



Fig. 33.9a,b The use of IADs for mail and package delivery service (a) and automobile industries (b)

ation of awkward and uncomfortable handling situations for mail handlers:

- the heavy weight of the sacks and letter trays and letter tubs
- the lack of handles, eyelets or any other helpful operator interface on the sacks and parcels
- the unpredictable shape, size, and weight of the sacks and letter trays and letter tubs at a work station

Intelligent assist devices greatly reduce the risk of back injuries when used by workers performing repetitive maneuvers. This reduction in injury, in turn, will greatly reduce the national cost of treating back injuries. See [33.35] and [33.36] for end-effectors that are designed for grasping postal sacks.

33.6 Lower-Extremity Exoskeleton

The first field-operational lower-extremity exoskeleton (commonly referred to as BLEEX) is comprised of two powered anthropomorphic legs, a power unit, and

a backpack-like frame on which a variety of heavy loads can be mounted. This system provides its pilot (i. e., the wearer) with the ability to carry significant loads

on his/her back with minimal effort over any type of terrain. BLEEX allows the pilot to comfortably squat, bend, swing from side to side, twist, and walk on ascending and descending slopes, while also offering the ability to step over and under obstructions while carrying equipment and supplies. Because the pilot can carry significant loads for extended periods of time without reducing his/her agility, physical effectiveness increases significantly with the aid of this class of lower-extremity exoskeletons. In order to address issues of field robustness and reliability, BLEEX is designed such that, in the case of power loss (e.g., from fuel exhaustion), the exoskeleton legs can be easily removed and the remainder of the device can be carried like a standard backpack.

BLEEX was first unveiled in 2004, at UC Berkeley's Human Engineering and Robotics Laboratory. In this initial model, BLEEX offered a carrying capacity of 34 kg (75 lbs), with weight in excess of that allowance being supported by the pilot. BLEEX's unique design offers an ergonomic, highly maneuverable, mechanically robust, lightweight, and durable outfit to

surpass typical human limitations. BLEEX has numerous potential applications; it can provide soldiers, disaster relief workers, wildfire fighters, and other emergency personnel with the ability to carry heavy loads such as food, rescue equipment, first-aid supplies, communications gear, and weaponry, without the strain typically associated with demanding labor. Unlike unrealistic fantasy-type concepts fueled by movie-makers and science-fiction writers, the lower-extremity exoskeleton conceived at Berkeley is a practical, intelligent, load-carrying robotic device. It is our vision that BLEEX will provide a versatile and realizable transport platform for mission-critical equipment.

The effectiveness of the lower-extremity exoskeleton stems from the combined benefit of the human intellect provided by the pilot and the strength advantage offered by the exoskeleton; in other words, the human provides an intelligent control system for the exoskeleton while the exoskeleton actuators provide most of the strength necessary for walking. The control algorithm ensures that the exoskeleton moves in concert with the pilot with minimal interaction force between the two. The control scheme needs no direct measurements from the pilot or the human-machine interface (e.g., no force sensors between the two); instead, the controller estimates, based on measurements from the exoskeleton only, how to move so that the pilot feels very little force. This control scheme, which has never before been applied to any robotic system, is an effective method of generating locomotion when the contact location between the pilot and the exoskeleton is unknown and unpredictable (i.e., the exoskeleton and the pilot are in contact in variety of places). This control method differs from compliance control methods [33.27,28] employed for upper-extremity exoskeletons [33.17,21] and haptic systems [33.18,19] because it requires no force sensor between the wearer and the exoskeleton.

The basic principle for the control of an exoskeleton rests on the notion that the exoskeleton needs to shadow the wearer's voluntary and involuntary movements quickly, and without delay. This requires a high level of sensitivity in response to all forces and torques on the exoskeleton, particularly the forces imposed by the pilot. Addressing this need involves a direct conflict with control science's goal of minimizing system sensitivity in the design of a closed-loop feedback system. If fitted with a low sensitivity, the exoskeleton would not move in concert with its wearer. One should realize, however, that maximizing system sensitivity to external forces and torques leads to a loss of robustness in the system.



Fig. 33.10 Berkeley lower-extremity exoskeleton (BLEEX) and pilot Ryan Steger. 1: The load occupies the upper portion of the backpack and around the power unit; 2: rigid connection of the BLEEX spine to the pilot's vest; 3: the power unit and central computer occupies the lower portion of the backpack; 4: semirigid vest connecting BLEEX to the pilot; 5: one of the hydraulic actuators; 6: rigid connection of the BLEEX feet to the pilot's boots

Taking into account this new approach, the goal is to develop a controller for the exoskeleton with high sensitivity. One is faced with two realistic concerns; the first was that an exoskeleton with high sensitivity to external forces and torques would respond to other external forces not initiated by its pilot, for example, if someone pushed against an exoskeleton that had high sensitivity, the exoskeleton would move just as it would in response to forces from its pilot. Although the fact that it does not stabilize its behavior on its own in response to other forces may sound like a serious problem, if it did (e.g., using a gyro) the pilot would receive motion from the exoskeleton unexpectedly and would have to struggle with it to avoid unwanted movement. The key to

stabilizing the exoskeleton and preventing it from falling in response to external forces depends on the pilot's ability to move quickly (e.g., step back or sideways) to create a stable situation for himself and the exoskeleton. For this, a very wide control bandwidth is needed so that the exoskeleton can respond to both pilot's voluntary and involuntary movements (i. e., reflexes).

The second concern is that systems with high sensitivity to external forces and torques are not robust to variations and therefore the precision of the system performance will be proportional to the precision of the exoskeleton dynamic model. Various experimental systems in Berkeley have proved the overall effectiveness of the control method in shadowing the pilot's movement.

33.7 The Control Scheme of an Exoskeleton

The control of the exoskeleton is motivated here through the simple one-degree-of-freedom (1-DOF) example shown in Fig. 33.11. This figure schematically depicts a human leg attached or interacting with a 1-DOF exoskeleton leg in a swing configuration (no interaction with the ground). For simplicity, the exoskeleton leg is shown as a rigid link pivoting about a joint and powered by a single actuator. The exoskeleton leg in this example has an actuator that produces a torque about pivot point A.

Although the pilot is securely attached to the exoskeleton at the foot, other parts of the pilot leg, such as the shanks and thighs, can contact the exoskeleton and impose forces and torques on the exoskeleton leg. The location of the contacts and the direction of the contact forces (and sometimes contact torques) vary and are

therefore considered unknown values in this analysis. In fact, one of the primary objectives in designing BLEEX was to ensure a pilot's unrestricted interaction with the exoskeleton. The equivalent torque on the exoskeleton leg, resulting from the pilot's applied forces and torques, is represented by d .

In the absence of gravity, (33.9) and the block diagram of Fig. 33.12 represent the dynamic behavior of the exoskeleton leg regardless of any kind of internal feedback the actuator may have

$$\dot{v} = Gr + Sd, \quad (33.9)$$

where G represents the transfer function from the actuator input r to the exoskeleton angular velocity v (the actuator dynamics are included in G). In the case where multiple actuators produce controlled torques on the system, r is the vector of torques imposed on the exoskeleton by the actuators. The form of G and the type of internal feedback for the actuator is immaterial for the discussion here. Also bear in mind the omission of the Laplace operator in all equations for the sake of compactness.

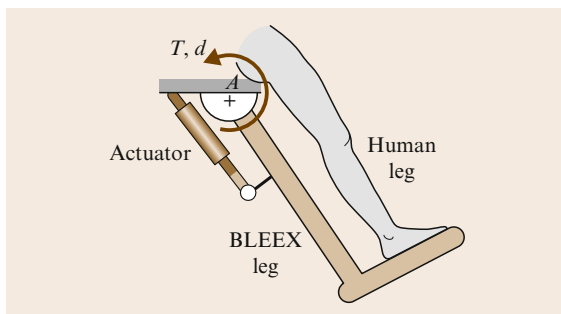


Fig. 33.11 Simple one-DOF exoskeleton leg interacting with the pilot leg. The exoskeleton leg has an actuator that produces a torque T about the pivot point A. The total equivalent torque associated with all forces and torques from the pilot on the exoskeleton is represented by d

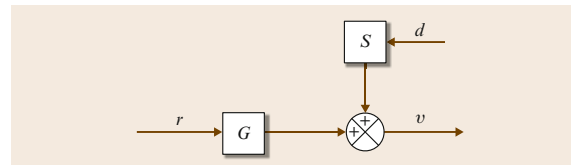


Fig. 33.12 The exoskeleton's angular velocity is shown as a function of the input to the actuators and the torques imposed by the pilot on the exoskeleton

The exoskeleton velocity, as shown by (33.9), is affected by forces and torques from its pilot. The sensitivity transfer function, S , represents how the equivalent human torque affects the exoskeleton angular velocity; S maps the equivalent pilot torque d onto the exoskeleton velocity v . If the actuator already has some sort of primary stabilizing controller, the magnitude of S will be small and the exoskeleton will only have a small response to the imposed forces and torques from the pilot or any other source. For example, a high-gain velocity controller in the actuator results in small S , and consequently a small exoskeleton response to forces and torques. Also, non-back-drivable actuators (e.g., large transmission ratios or servo-valves with overlapping spools) result in a small S , which leads to a correspondingly small response to pilot forces and torques.

Note that d (the resulting torque from pilot on the exoskeleton) is not an exogenous input; it is a function of the pilot dynamics and variables such as position and velocity of the pilot and the exoskeleton legs. These dynamics change from person to person, and within a person as a function of time and posture. It is assumed that d is only from the pilot and does not include any other external forces and torques.

The objective is to increase exoskeleton sensitivity to pilot forces and torques through feedback but without measuring d . Measuring d to create such systems develops several hard, but ultimately solvable, problems in the control of a lower-extremity exoskeleton. Some of those problems are briefly described below.

1. Depending on the architecture and the design of the exoskeleton, one needs to install several force and torque sensors to measure all forces from the pilot on the exoskeleton because the pilot is in contact with the exoskeleton at several locations. These locations are not known in advance. For example, we have found that some pilots are interested in having braces connecting an exoskeleton at the shanks while some are interested in having them on the thighs. Inclusion of sensors on a leg to measure all kinds of human forces and torques may result in a system suitable for a laboratory setting but not robust enough to be deployed in the field.
2. If the exoskeleton design is such that the forces and torques applied by the pilot on the exoskeleton are limited to a specified location (e.g., the pilot foot), the sensor that measures the pilot forces and torques will also inadvertently measure other forces and torques that are not intended for locomotion. This is a major difference between measuring forces

from, for example, the human hands, and measuring forces from the human lower limbs. Using our hands, we are able to impose controlled forces and torques on upper extremity exoskeletons and haptic systems with very few uncertainties. However, our lower limbs have other primary and nonvoluntary functions like load support that take priority over locomotion.

3. One option which was experimented with was the installation of sensing devices for forces on the bottom of the pilot's boots, where they are connected to the exoskeleton. Since the force on the bottom of the pilot's boot travels from heel to toe during normal walking, several sensors are required to measure the pilot force. Ideally, one would have a matrix of force sensors between the pilot and exoskeleton feet to measure the pilot forces at all locations and at all directions, though in practice, only a few sensors could be accommodated: at the toe, ball, midfoot, and the heel. Still, this option leads to thick and bulky soles.
4. The bottoms of the human boots experience cyclic forces and torques during normal walking that lead to fatigue and eventual sensor failure if the sensor is not designed and isolated properly.

For the above reasons and our experience in the design of various lower-extremity exoskeletons, it became evident that the existing state of technology in force sensing could not provide robust and repeatable measurement of the human lower limb force on the exoskeleton. Our goal then shifted to developing an exoskeleton with a large sensitivity to forces and torques from the operator using measurements only from the exoskeleton (i.e., no sensors on the pilot or the exoskeleton interface with the pilot). Creating a feedback loop from the exoskeleton variables only, as shown in Fig. 33.13 the new closed-loop sensitivity transfer function is presented in (33.10).

$$S_{\text{NEW}} = \frac{v}{d} = \frac{S}{1 + GC} \quad (33.10)$$

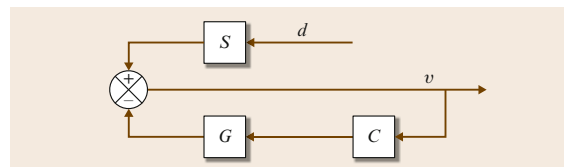


Fig. 33.13 The feedback control loop added to block diagram of Fig. 33.12; C is the controller operating only on the exoskeleton variables

Observation of (33.10) reveals that $S_{\text{NEW}} \leq S$, and therefore any negative feedback from the exoskeleton leads to an even smaller sensitivity transfer function. With respect to (33.10), our goal is to design a controller for a given S and G such that the closed-loop response from d to v (the new sensitivity function as given by (33.10)) is greater than the open-loop sensitivity transfer function (i.e., S) within some bounded frequency range. This design specification is given by inequality (33.11)

$$|S_{\text{NEW}}| > |S|, \quad \forall \omega \in (0, \omega_0), \quad (33.11)$$

or alternatively

$$|1 + GC| < 1 \quad \forall \omega \in (0, \omega_0), \quad (33.12)$$

where ω_0 is the exoskeleton maneuvering bandwidth.

In classical and modern control theory, every effort is made to minimize the sensitivity function of a system to external forces and torques. But for exoskeleton control, one requires a totally opposite goal: *to maximize the sensitivity of the closed-loop system to forces and torques*. In classical servo problems, negative feedback loops with large gains generally lead to small sensitivity within a bandwidth, which means that they reject forces and torques (usually called disturbances). However, the above analysis states that the exoskeleton controller needs a large sensitivity to forces and torques.

To achieve a large sensitivity function, it is suggested that one uses the inverse of the exoskeleton dynamics as a positive feedback controller so that the loop gain for the exoskeleton approaches unity (slightly less than 1). Assuming positive feedback, (33.10) can be written as

$$S_{\text{NEW}} = \frac{v}{d} = \frac{S}{1 - GC}. \quad (33.13)$$

If C is chosen to be $C = 0.9G^{-1}$, then the new sensitivity transfer function is $S_{\text{NEW}} = 10S$ (ten times the force amplification). In general we recommend the use of positive feedback with a controller chosen as:

$$C(1 - \alpha^{-1})G^{-1}, \quad (33.14)$$

where α is the amplification number greater than unity (for the above example, $\alpha = 10$ led to the choice of $C = 0.9G^{-1}$). Equation (33.14) simply states that a positive-feedback controller needs to be chosen as the inverse dynamics of the system dynamics scaled down by $(1 - \alpha^{-1})$. Note that (33.14) prescribes the controller in the absence of unmodeled high-frequency exoskeleton dynamics. In practice, C also includes a unity-gain low-pass filter to attenuate the unmodeled high-frequency exoskeleton dynamics.

The above method works well if the system model (i.e., G) is well known to the designer. If the model is not well known, then the system performance will differ greatly from the one predicted by (33.13), and in some cases instability will occur. The above simple solution comes with an expensive price: robustness to parameter variations. In order to get the above method working, one needs to know the dynamics of the system well, to understand the dynamics of the exoskeleton quite well, as the controller is heavily model based. One can see this problem as a tradeoff: the design approach described above requires no sensor (e.g., force or EMG) in the interface between the pilot and the exoskeleton; one can push and pull against the exoskeleton in any direction and at any location without measuring any variables on the interface. However, the control method requires a very good model of the system. At this time, the experiments with the exoskeleton have shown that this control scheme – which does not stabilize the exoskeleton – forces the system to follow wide-bandwidth human maneuvers while carrying heavy loads. We have come to believe, to rephrase Friedrich Nietzsche, that *that which does not stabilize, will only make us stronger*. Reference [33.37] describes a system identification method for BLEEX.

How does the pilot dynamic behavior affect the exoskeleton behavior? In our control scheme, there is no need to include the internal components of the pilot limb model; the detailed dynamics of nerve conduction, muscle contraction, and central nervous system processing are implicitly accounted for in constructing the dynamic model of the pilot limbs. The pilot force on the exoskeleton, d , is a function of both the pilot dynamics, H , and the kinematics of the pilot limb (e.g., velocity, position or a combination thereof). In general, H is determined primarily by the physical properties of the human dynamics. Here it is assumed H is a nonlinear operator representing the pilot impedance as a function of the pilot kinematics

$$d = -H(v). \quad (33.15)$$

The specific form of H is not known other than that it results in the human muscle force on the exoskeleton. Figure 33.14 represents the closed-loop system behavior when pilot dynamics is added to the block diagram of Fig. 33.13. Examining Fig. 33.14 reveals that (33.13), representing the new exoskeleton sensitivity function, is not affected by the feedback loop containing H .

Figure 33.14 shows an important characteristic of exoskeleton control. One can observe two feedback loops in the system. The upper feedback loop represents

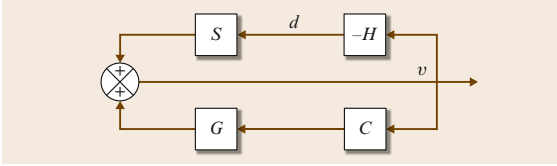


Fig. 33.14 Block diagram showing how an exoskeleton moves. The *upper loop* shows how the pilot moves the exoskeleton through applied forces. The *lower loop* shows how the controller drives the exoskeleton

how forces and torques from the pilot affect the exoskeleton. The lower loop shows how the controlled feedback loop affects the exoskeleton. While the lower feedback loop is positive (potentially destabilizing), the upper feedback loop stabilizes the overall system of pilot and exoskeleton taken as a whole. See [33.38] for detailed stability analysis where it can be seen that, unlike control methods utilized in the control of upper-extremity exoskeletons [33.19], the human dynamics in the control method described here has little potential to destabilize the system. Even though the feedback loop containing C is positive, the feedback loop containing H stabilizes the overall system of pilot and exoskeleton.

The above discussion motivated the design philosophy using a 1-DOF system. An exoskeleton, as shown is a system with many degrees of freedom and therefore implementation of the controller needs further attention. Below we extend the above control technique to the single support phase only. Refer to [33.38] and [33.39] for more details associated with multivariable control.

In the single support phase, the exoskeleton system is modeled as the seven-DOF serial link mechanism in the sagittal plane shown in Fig. 33.15. The inverse dynamics of the exoskeleton can be written in the general form as:

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + P(\theta) = T + d, \quad (33.16)$$

where $\theta = (\theta_1 \theta_2 \dots \theta_7)^\top$ and $T = (0, T_1 T_2 \dots T_6)^\top$.

M is a 7×7 inertia matrix and is a function of θ , $C(\theta, \dot{\theta})$ is a 7×7 centripetal and Coriolis matrix and is a function of θ and $\dot{\theta}$, and P is a 7×1 vector of gravitational torques and is a function of θ only. T is the 7×1 actuator torque vector with its first element set to zero since there is no actuator associated with joint angle θ_1 (i.e., angle between the exoskeleton foot and the ground); d is the effective 7×1 torque vector imposed by the pilot on the exoskeleton at various locations. According to (33.14), we choose the controller to be the exoskeleton inverse dynamics scaled by $(1 - \alpha^{-1})$, where α is

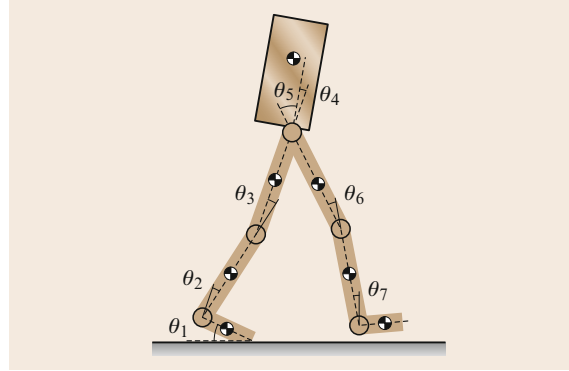


Fig. 33.15 Sagittal plane representation of the exoskeleton in the single stance phase. The *torso* includes the combined exoskeleton torso mechanism, payload, control computer, and power source

the amplification number

$$T = \hat{P} + (1 - \alpha^{-1})[\hat{M}(\theta)\ddot{\theta} + \hat{C}(\theta, \dot{\theta})\dot{\theta}], \quad (33.17)$$

$\hat{C}(\theta, \dot{\theta})$, $\hat{P}(\theta)$ and $\hat{M}(\theta)$ are the estimates of the Coriolis matrix, gravity vector, and the inertia matrix, respectively, for the system shown in Fig. 33.15. Note that (33.17) results in a 7×1 actuator torque. Since there is no actuator between the exoskeleton foot and the ground, the torque prescribed by the first element of T must be provided by the pilot. Substituting T from (33.17) into (33.16) yields,

$$\begin{aligned} M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + P(\theta) \\ = \hat{P}(\theta) + (1 - \alpha^{-1})[\hat{M}(\theta)\ddot{\theta} + \hat{C}(\theta, \dot{\theta})\dot{\theta}] + d. \end{aligned} \quad (33.18)$$

In the limit when $M(\theta) = \hat{M}(\theta)$, $C(\theta, \dot{\theta}) = \hat{C}(\theta, \dot{\theta})$, $P(\theta) = \hat{P}(\theta)$, and α are sufficiently large, d will approach zero, meaning the pilot can walk as if the exoskeleton did not exist. However, it can be seen from (33.18) that the force felt by the pilot is a function of α and the accuracy of the estimates $\hat{C}(\theta, \dot{\theta})$, $\hat{P}(\theta)$, and $\hat{M}(\theta)$. In general, the more accurately the system is modeled, the less the human force, d , will be. In the presence of variations in abduction-adduction angles, only $P(\theta)$ in (33.16) and (33.17) needs to be modified.

Exoskeleton systems use multivariable nonlinear algorithm to robustly control their behavior. Since all computations required to implement the control are conducted on a single computer, one needs a control platform to minimize the number of signal wires in the system. See [33.40] and [33.41] for a novel control platform.

33.8 Highlights of the Lower-Extremity Design

In designing an exoskeleton, several factors had to be considered: Firstly, the exoskeleton needed to exist in the same workspace of the pilot without interfering with his motion. Secondly, it had to be decided whether the exoskeleton should be anthropomorphic (i. e., kinematically

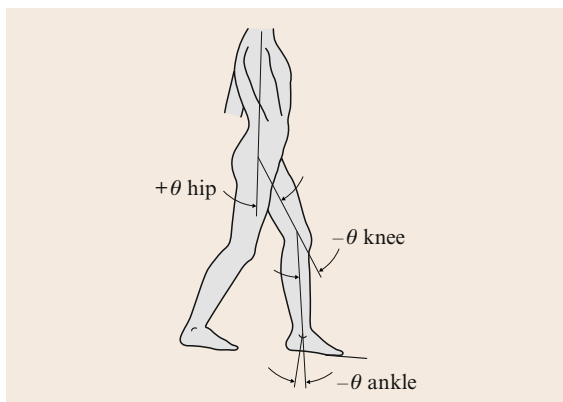


Fig. 33.16 Each joint angle is measured as the positive counterclockwise displacement of the distal link from the proximal link (zero in standing position) with the person oriented as shown. In the position shown, the hip angle is positive whereas both the knee and ankle angles are negative. Torque is measured as positive acting counterclockwise on the distal link

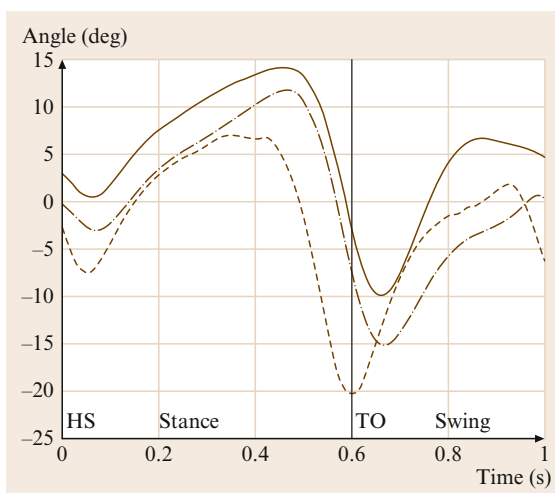


Fig. 33.17 Three sets of adjusted CGA data of the ankle flexion/extension angle. The minimum angle (extension) is $\approx -20^\circ$ and occurs just after toe-off. The maximum angle (flexion) is $\approx +15^\circ$ and occurs in late stance phase

matching), or non-anthropomorphic (i. e., kinematically matching the operator only at the connection points between human and machine). Berkeley ultimately selected the anthropomorphic architecture because of its transparency to the pilot. It is also concluded that an exoskeleton that kinematically matches the wearer's legs gains the most psychological acceptance by the user and is therefore safer to wear. Consequently, the exoskeleton was designed to have the same degrees of freedom as the pilot: three degrees at the ankle and the hip, and one degree at the knee. This architecture also allowed the appropriately scaled clinical human walking data to be employed for the design of the exoskeleton components, including the workspace, actuators, and the power source.

A study of clinical gait analysis (CGA) data provides evidence that humans expend the most power through the sagittal plane joints of the ankle, knee, and hip while walking, squatting, climbing stairs, and most other common maneuvers. For this reason, the sagittal-plane joints of the first prototype exoskeleton are powered. However, to save energy, the nonsagittal degrees of freedom at the ankle and hip remain unpowered. This compels the pilot to provide the force to maneuver the exoskeleton abduction and rotation, where the required operational forces are smaller. To reduce the burden on the human operator further, the unactuated degrees of freedom are spring loaded to a neutral standing position.

Human joint angles and torques for a typical walking cycle were obtained in the form of independently collected CGA data. CGA angle data is typically collected via human video motion capture. CGA torque data is calculated by estimating limb masses and inertias and applying dynamic equations to the motion data. Given the variations in individual gait and measuring methods, three independent sources of CGA data [33.42–44] were utilized for the analysis and design of BLEEX. This data was modified to yield estimates of exoskeleton actuation requirements. The modifications included: (1) scaling the joint torques to a 75 kg person (the projected weight of the exoskeleton and its payload not including its pilot); (2) scaling the data to represent the walking speed of one cycle per second (or about 1.3 m/s); and (3) adding the pelvic tilt angle (or lower back angle depending on data available) to the hip angle to yield a single hip angle between the torso and the thigh, as shown in Fig. 33.16. This accounts for the reduced degrees of freedom of the exoskeleton. The following sections describe the use of CGA data and its implication for the exoskeleton design. The sign conventions used are shown in Fig. 33.16.

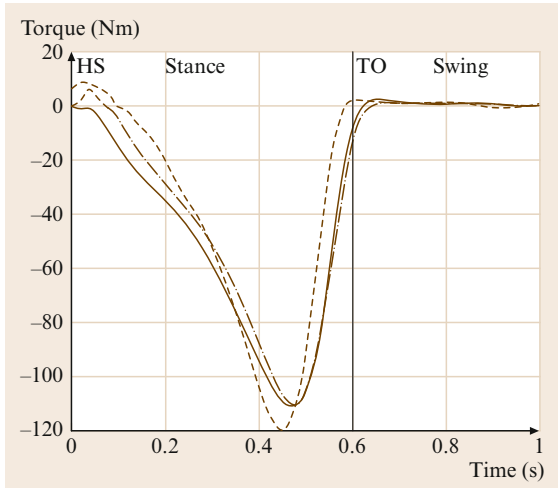


Fig. 33.18 Three sets of adjusted CGA data of the ankle flexion/extension torque. Peak negative torque (extension of the foot) is very large (≈ -120 Nm) and occurs in late stance phase. The ankle torque during swing is quite small

Figure 33.17 shows the CGA ankle angle data for a 75 kg human walking on flat ground at approximately 1.3 m/s versus time. Although Fig. 33.17 shows a small range of motion while walking (approximately -20° to $+15^\circ$), greater ranges of motion are required for other movements. An average person can flex their ankles anywhere from -38° to $+35^\circ$. The exoskeleton ankle was chosen to have a maximum flexibility of $\pm 45^\circ$ to compensate for the lack of several smaller degrees of freedom in the exoskeleton foot. Through all plots, TO stands for *toe-off* and HS stands for *heel-strike*.

Figure 33.18 shows the adjusted CGA data of the ankle flexion/extension torque. The ankle torque is almost entirely negative, making unidirectional actuators an ideal actuation choice. This asymmetry also implies a preferred mounting orientation for asymmetric actuators (one sided hydraulic cylinders). Conversely, if symmetric bidirectional actuators are considered, spring loading would allow the use of low-torque-producing actuators. Although the ankle torque is large during stance, it is negligible during swing. This suggests a system that disengages the ankle actuators from the exoskeleton during swing to save power.

The instantaneous ankle mechanical power (shown in Fig. 33.19) is calculated by multiplying the joint angular velocity (derived from Fig. 33.17) and the instantaneous joint torque (Fig. 33.18). The ankle absorbs energy during the first half of the stance phase and releases energy just before toe off. The average ankle

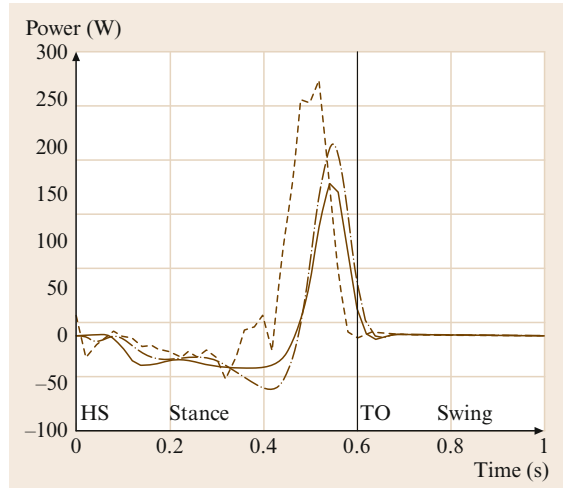


Fig. 33.19 Three sets of adjusted CGA data of the ankle flexion/extension instantaneous mechanical power. The average ankle power is positive, indicating that the ankle does positive work and requires actuation

power is positive, indicating that power production is required at the ankle.

Similar analyses were carried out for the knee and the hip [33.45] and [33.46]. The required knee torque has both positive and negative components, indicating the need for a bidirectional actuator. The highest peak torque is extension in early stance (≈ 60 Nm); hence asymmetric actuators should be biased to provide greater extension torque. The hip torque is relatively symmetric (-80 to $+60$ Nm); hence a bidirectional hip actuator is required. Negative extension torque is required in early stance as the hip supports the load on the stance leg.

Hip torque is positive in late stance and early swing as the hip propels the leg forward during swing. In late swing, the torque goes negative as the hip decelerates the leg prior to heel strike.

CGA data, which provided torque and speed information at each joint of a 75 kg person, was also used to size the exoskeleton power source. The information suggested that a typical person uses about 0.25 HP (185 W) to walk at an average speed of 3 mph. This figure, which represents the average product of speed and torque, is an expression of the purely mechanical power exhibited at the legs during walking. Since it is assumed that the exoskeleton is similar to a human in terms of geometry and weight, one of the key design objectives turned out to be designing a power unit and actuation system to deliver about 0.25 HP at the exoskeleton joints.

Table 33.1 BLEEX joint ranges of motion

	Human walking maximum (degrees)	BLEEX maximum (degrees)	Average military male maximum (degrees)
Ankle flexion	14.1	45	35
Ankle extension	20.6	45	38
Ankle abduction	not available	20	23
Ankle adduction	not available	20	24
Knee flexion	73.5	121	159
Hip flexion	32.2	121	125
Hip extension	22.5	10	not available
Hip abduction	7.9	16	53
Hip adduction	6.4	16	31
Total rotation external	13.2	35	73
Total rotation internal	1.6	35	66

The BLEEX kinematics are close to human leg kinematics, so the BLEEX joint ranges of motion are determined by examining human joint ranges of motion. At the very least, the BLEEX joint range of motion should be equal to the human range of motion during walking (shown in column 1 in Table 33.1), which can be found by examining CGA data [33.42–44]. Safety dictates that the BLEEX range of motion should not be more than the operator’s range of motion (shown in column 3 of Table 33.1). For each degree of freedom, the second column of Table 33.1 lists the BLEEX range of motion which is in general larger than the human range of motion during walking and less than the maximum range of human motion.

The most maneuverable exoskeleton should ideally have ranges of motion slightly less than the human’s maximum range of motion. However, BLEEX uses linear actuators, so some of the joint ranges of motion are reduced to prevent the actuators’ axes of motion from passing through the joint center. If this had not been prevented, the joint could reach a configuration where the actuator would be unable to produce a torque about its joint. Additionally, all the joint ranges of motion were tested and revised during prototype testing. For example, mockup testing determined that the BLEEX ankle flexion/extension range of motion needs to be greater than the human ankle range of motion to accommodate the human foot’s smaller degrees of freedom not modeled in the BLEEX foot.

It is natural to design a 3-DOF exoskeleton hip joint such that all three axes of rotation pass through the human ball-and-socket hip joint. However, through the design of several mockups and experiments, we learned that these designs have limited ranges of motion and result in singularities at some human hip postures.

Therefore the rotation joint was moved so it does not align with the human’s hip joint. Initially the rotation joint was placed directly above each exoskeleton leg (labeled ‘alternate rotation’ in Fig. 33.20). This worked well for the lightweight plastic mockup, but created problems in the full-scale prototype because the high mass of the torso and payload created a large moment about the unactuated rotation joint. Therefore, the current hip rotation joint for both legs was chosen to be a single axis of rotation directly behind the person and under the torso (labeled ‘current rotation’ in Fig. 33.20). The current rotation joint is typically spring loaded towards its illustrated position using sheets of spring steel.

Like the human’s ankle, the BLEEX ankle has three DOFs. The flexion/extension axis coincides with the human ankle joint. For design simplification, the abduction/adduction and rotation axes on the BLEEX ankle do not pass through the human’s leg and form a plane outside of the human’s foot (Fig. 33.21). To take load

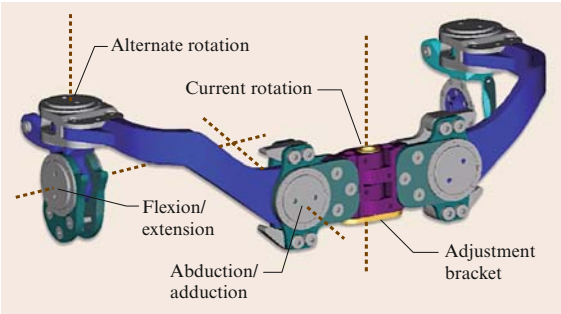


Fig. 33.20 The exoskeleton hip degrees of freedom (back view). Only the rotation axis does not pass through the human hip ball and socket joint. The adjustment bracket is replaceable to accommodate various sized operators

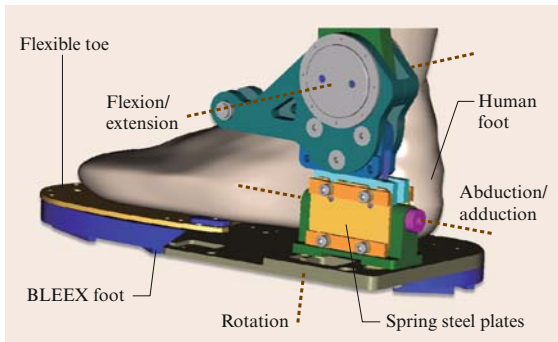


Fig. 33.21 The exoskeleton ankle degrees of freedom. Only the flexion/extension axis passes through the human's ankle joint. Abduction/adduction and rotation axes are not powered, but are equipped with appropriate impedances

off of the human's ankle, the BLEEX ankle abduction/adduction joint is sprung towards vertical, but the rotation joint is completely free. Additionally, the front of the exoskeleton foot, under the operator's toes, is compliant to allow the exoskeleton foot to flex with the human's foot. Since the human and exoskeleton leg kinematics are not exactly the same (merely similar), the human and exoskeleton are only rigidly connected at the extremities (feet and torso).

The BLEEX foot is a critical component due to its variety of functions.

- It measures the location of the foot's center of pressure and therefore identifies the foot's configuration on the ground. This information is necessary for BLEEX control.
- It measures the human's load distribution (how much of the human's weight is on each leg), which is also used in BLEEX control.

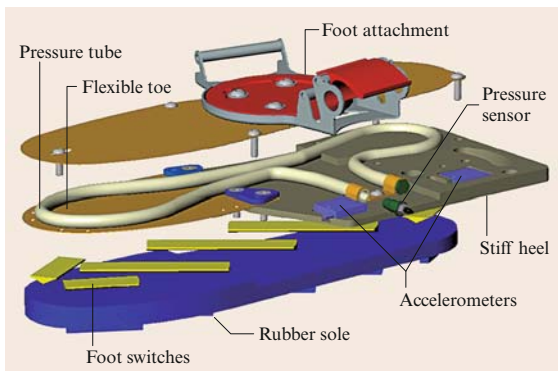


Fig. 33.22 BLEEX foot design (exploded view)

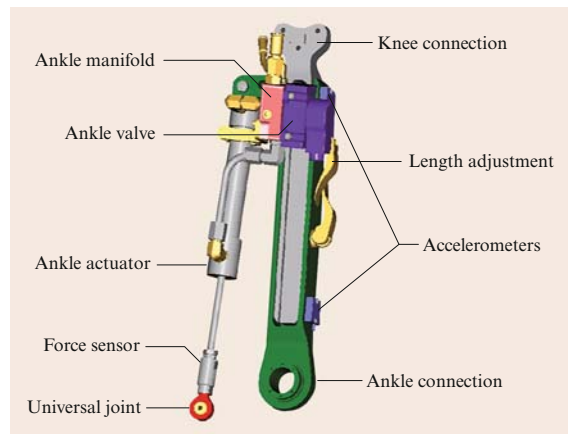


Fig. 33.23 BLEEX shank design

- It transfers BLEEX's weight to the ground, so it must have structural integrity and exhibit long life in the presence of periodic environmental forces.
- It is one of two places where the human and exoskeleton are rigidly connected, so it must be comfortable for the operator.

As shown in Fig. 33.21, the main structure of the foot has a stiff heel to transfer the load to the ground and a flexible toe for comfort. The operator's boot rigidly attaches to the top of the exoskeleton foot via a quick-release binding. Along the bottom of the foot, switches detect which parts of the foot are in contact with the ground. For ruggedness, these switches are molded into a custom rubber sole. Also illustrated in Fig. 33.21 is the load distribution sensor, a rubber *pressure tube* filled with hydraulic oil and sandwiched between the human's

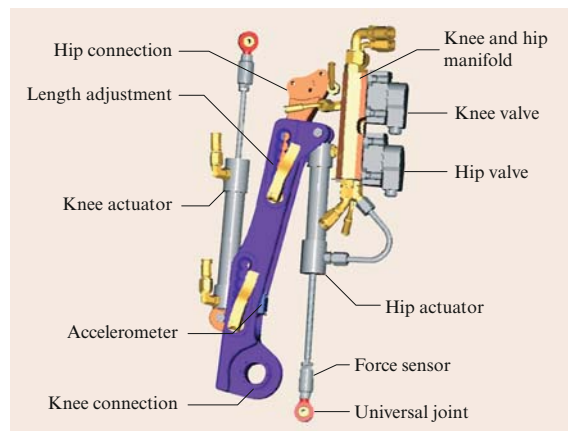


Fig. 33.24 BLEEX thigh design

foot and the main exoskeleton foot structure. Only the weight of the human (not the exoskeleton) is transferred onto the pressure tube and measured by the sensor. This sensor is used by the control algorithm to detect how much weight the human places on their left leg versus their right leg.

The main function of the BLEEX shank and thigh are for structural support and to connect the flexion/extension joints together (Figs. 33.23 and 33.24). Both the shank and thigh are designed to adjust to fit 90% of the population; they consist of two pieces that slide within each other and then lock at the desired length. To minimize the hydraulic routing, manifolds were designed to route the fluid between the valves, actuators, supply, and return lines. These manifolds mount directly to the cylinders to reduce the hydraulic distance between the valves and actuator, maximizing the actua-

tor's performance. The actuator, manifold, and valve for the ankle mount to the shank, while the actuators, manifold, and valves for the knee and hip are on the thigh. One manifold, mounted on the knee actuator, routes the hydraulic fluid for the knee and hip actuators.

Shown in Fig. 33.26, the BLEEX torso connects to the hip structure (shown in Fig. 33.20). The power supply [33.47–49], controlling computer, and payload mount to the rear side of the torso. An inclinometer mounted to the torso gives the absolute angle reference for the control algorithm. A custom harness (Fig. 33.27) mounts to the front of the torso to hold the exoskeleton to the operator. Besides the feet, the harness is the only other location where the user and exoskeleton are rigidly connected. Figure 33.26 also illustrates the actuator, valve, and manifold for the hip abduction/adduction joint.

33.9 Field-Ready Exoskeleton Systems

This section describes two field-ready exoskeletons developed by members of Berkeley Bionics in conjunction with researchers from the University of California.

33.9.1 The ExoHiker Exoskeleton

The ExoHiker exoskeleton (shown in Fig. 33.25) was the first exoskeleton created by the team of Berkeley Bionics and the University of California, and was the first human exoskeleton in the world capable of rigorous customer evaluations as a load-carrying device. It was designed for carrying heavy backpacks for long missions with small changes in altitude. It weighs only 14.5 kg including a power unit and onboard computer. Its payload capacity is 90 kg while the wearer feels negligible load. The noise from this device is imperceptible. A production version of such an exoskeleton would be capable of traveling 68 km with 0.5 kg of batteries (lithium polymer) at an average speed 4.0 km/h while carrying a 68 kg backpack. When supplemented with a small solar panel, its mission time could be unlimited. The ExoHiker is adjustable to fit individuals ranging in height from approximately 1.65 m to 1.91 m tall. All control adjustments can be accessed by the user through a simple handheld liquid-crystal display (LCD) controller equipped with a graphical user interface menu system. This exoskeleton has been evaluated by the Special Operations Research Support

Element on trails in the Rocky Mountains, and in the laboratory environment of the Natick Soldier Systems Center.



Fig. 33.25 The ExoHiker™ field-ready exoskeleton is suitable for small slopes and can carry a load of 90 kg (courtesy of Berkeley Bionics, Jan 2005)



Fig. 33.28 The ExoClimber™ field-ready exoskeleton is suitable for rapid ascent of stairs and steep slopes (courtesy of Berkeley Bionics, Oct 2005)

33.9.2 The ExoClimber Exoskeleton

This exoskeleton (shown in Fig. 33.28) is designed to allow rapid ascent of stairs and steep slopes while providing the same long term load carrying capability of ExoHiker. It weighs 23 kg including power source and onboard computer while its payload capacity is 90 kg. This exoskeleton is as loud as an office printer. Its battery requirements are the same as ExoHiker in all situations except ascending steep slopes. During a steep ascent, the ExoClimber is capable of ascending 300 m per 0.5 kg of battery added to the system (while carrying a 68 kg backpack). This exoskeleton has been evaluated by the Special Operations Research Support Element on trails in the Rocky Mountains, and in the laboratory environment of the Natick Soldier Systems Center. The evaluation included hiking on snow with and without snow shoes. The resulting report was extremely favorable, and during one experiment the distance an operator could walk with a 45 kg pack was increased by 900% using the ExoClimber.

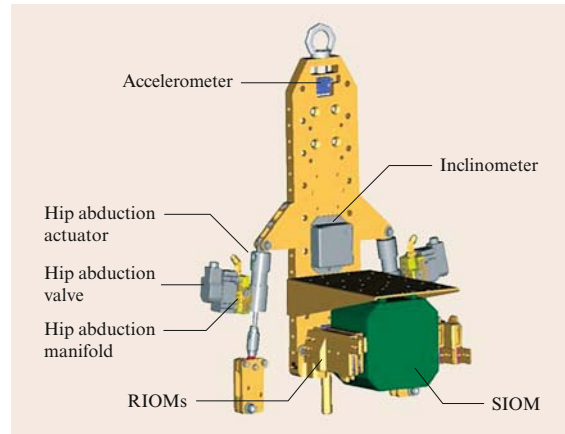


Fig. 33.26 BLEEX torso design (back view)

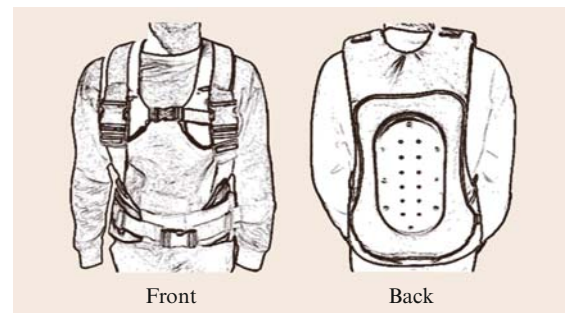


Fig. 33.27 The pilot vests in Fig. 33.10 are designed to distribute the BLEEX-pilot force uniformly on the pilot's upper body

The recent exoskeleton system from Berkeley Bionics now has two independent characteristics: (1) it increases its wearer's maximal load carrying capacity (68 kg to 90 kg), and (2) it decreases its wearer's metabolic cost. During some preliminary evaluations in late 2006 and early 2007, the oxygen consumption of the users walking at a speed of 3.2 km/h, was decreased by 5%–12% when using the latest exoskeleton without a payload. When the users carried a load, the effect was more pronounced. The oxygen consumption of these users carrying a load of 37 kg at a speed of 3.2 km/h was decreased by about 15% when using this exoskeleton. This is the very first exoskeleton in the world that has demonstrated a decrease in oxygen consumption of the user.

33.10 Conclusion and Further Reading

Exoskeleton technology is in its infancy. Although the exoskeleton systems in this analysis are generally considered robotics system worn by humans to carry loads, the reader should be aware that the field of medical orthotics also offers a wealth of knowledge and awareness associated with systems collaborating with patients either as rehabilitation devices or assist de-

vices. For examples of such systems, we invite the readers to see references [33.50, 51] associated with pneumatic ankle orthoses for rehabilitation. At this time, the most limiting issue in exoskeleton technology is the power supply and actuators. Without a viable power supply, exoskeleton systems will be limited to indoor applications.

References

- 33.1 N.J. Mizen: Preliminary design for the shoulders and arms of a powered, exoskeletal structure, Cornell Aeronaut. Lab. Rep. V0-1692-V-4 (1965)
- 33.2 General Electric Co.: Hardiman I Arm Test, General Electric Rep. S-70-1019, Schenectady (1969)
- 33.3 General Electric Co.: Hardiman I Prototype Project, Special Interim Study, General Electric Rep. S-68-1060, Schenectady (1968)
- 33.4 P.F. Groshaw, General Electric Co.: Hardiman I Arm Test, Hardiman I Prototype, General Electric Rep. S-70-1019, Schenectady (1969)
- 33.5 B.J. Makinson, General Electric Co.: Research and Development Prototype for Machine Augmentation of Human Strength and Endurance, Hardiman I Project, General Electric Rep. S-71-1056, Schenectady (1971)
- 33.6 R.S. Mosher: Force-reflecting electrohydraulic manipulator, Electro-Technology (1960)
- 33.7 M. Vukobratovic, V. Ciric, D. Hristic: Contribution to the Study of Active Exoskeletons, Proc. Of the 5th IFAC Congress (Paris, 1972)
- 33.8 D. Hristic, M. Vukobratovic: Development of active aids for handicapped, Proc. III International Conference on bio-Medical Engineering (Sorrento, 1973)
- 33.9 A. Seireg, J. Grundmann: *Design of a Multitask Exoskeletal Walking Device for Paraplegics, Biomechanics of Medical Devices* (Marcel Dekker, New York 1981) pp. 569-644
- 33.10 K. Yamamoto, K. Hyodo, M. Ishii, T. Matsuo: Development of power assisting suit for assisting nurse labor, JSME Int. J. Ser. C **45**(3), 703-711 (2002)
- 33.11 K. Yamamoto, M. Ishii, K. Hyodo, T. Yoshimitsu, T. Matsuo: Development of Power assisting suit, JSME Int. J. Ser. C **46**(3), 923-930 (2003)
- 33.12 J.E. Pratt, B.T. Krupp, C.J. Morse, S.H. Collins: The RoboKnee: An Exoskeleton for Enhancing Strength and Endurance During Walking, Proc. IEEE International Conference on Robotics and Automation (New Orleans, 2004) pp. 2430-2435
- 33.13 H. Kawamoto, Y. Sankai: Power assist system HAL-3 for gait disorder person, Int. Conf. Computer Helping People with Special Needs (Linz, 2002)
- 33.14 H. Kawamoto, S. Kanbe, Y. Sankai: Power assist method for HAL-3 estimating operator's intention based on motion information, Proc. of 2003 IEEE Workshop on Robot and Human Interactive Communication (Millbrae, 2003) pp. 67-72
- 33.15 K. Kong, D. Jeon: Design and control of an exoskeleton for the elderly and patients, IEEE/ASME Trans. Mechatron. **11**(4), 220-226 (2006)
- 33.16 S.K. Agrawal, A. Fattah: Theory and design of an orthotic device for full or partial gravity-balancing of a human leg during motion, IEEE Trans. Neural Syst. Rehab. Eng. **12**(2), 157-165 (2004)
- 33.17 H. Kazerooni, S. Mahoney: Dynamics and control of robotic systems worn by humans, ASME J. Dyn. Syst. Meas. Contr. **113**(3), 379-387 (1991)
- 33.18 H. Kazerooni, M. Her: The dynamics and control of a haptic interface device, IEEE Trans. Robot. Autom. **10**(4), 453-464 (1994)
- 33.19 H. Kazerooni, T. Snyder: A case study on dynamics of haptic devices: Human induced instability in powered hand controllers, AIAA J. Guid. Contr. Dyn. **18**(1), 108-113 (1995)
- 33.20 H. Kazerooni: Human-robot interaction via the transfer of power and information signals, IEEE Trans. Syst. Cybernet. **20**(2), 450-463 (1990)
- 33.21 H. Kazerooni, J. Guo: Human extenders, ASME J. Dyn. Syst. Meas. Contr. **115**(2B), 281-289 (1993)
- 33.22 H. Kazerooni: The extender technology at the University of California Berkeley, J. Soc. Instrum. Control Eng. Jpn. **34**, 291-298 (1995)
- 33.23 H. Kazerooni: The human power amplifier technology at the University of California Berkeley, J. Robot. Auton. Syst. **19**, 179-187 (1996)
- 33.24 U. Yutaka, H. Kazerooni: A μ -based synthesis based control for compliant maneuver, IEEE Conference on Systems, Man, and Cybernetics (Tokyo, 1999) pp. 1014-1019
- 33.25 T.J. Snyder, H. Kazerooni: A novel material handling system, IEEE International Conference on Robotics and Automation (1996) pp. 1147-1152
- 33.26 H. Kazerooni, T.B. Sheridan, P.K. Houpt: Robust compliant motion for manipulators, Part I: The

- fundamental concepts of compliant motion, *IEEE J. Robot. Autom.* **2**(2), 83–92 (1986)
- 33.27 H. Kazerooni: On the robot compliant motion control, *ASME J. Dyn. Syst. Meas. Contr.* **111**(3), 416–425 (1989)
- 33.28 H. Kazerooni, B.J. Waibel, S. Kim: On the stability of robot compliant motion control: Theory and experiments, *ASME J. Dyn. Syst. Meas. Contr.* **112**(3), 417–426 (1990)
- 33.29 H. Kazerooni, B.J. Waibel: Theories and experiments on the stability of robot compliance control, *IEEE Trans. Robot. Autom.* **7**(1), 95–104 (1991)
- 33.30 Pneumatic human power amplifier module, Patent #5,915,673
- 33.31 Human power amplifier for vertical maneuvers (U.S. Patent #5,865,426)
- 33.32 Human power amplifier for lifting load with slack prevention apparatus (U.S. Patent #6,622,990)
- 33.33 Device and Method for Wireless Lifting Assist Device (U.S. Patent #6,681,638)
- 33.34 H. Kazerooni, D. Fairbanks, A. Chen, G. Shin: The Magic Glove, *IEEE International Conference on Robotics and Automation* (New Orleans, 2004)
- 33.35 H. Kazerooni, C. Foley: A practical robotic end-effector for grasping postal sacks, *ASME J. Dyn. Syst. Meas. Contr.* **126**, 154–161 (2004)
- 33.36 H. Kazerooni, C. Foley: A robotic mechanism for grasping sacks, *IEEE Trans. Autom. Sci. Eng.* **2**(2), 111–120 (2005)
- 33.37 J. Ghan, R. Steger, H. Kazerooni: Control and system identification for the Berkeley lower extremity exoskeleton, *Adv. Robot.* **20**(9), 989–1014 (2006)
- 33.38 H. Kazerooni, R. Steger: The Berkeley lower extremity exoskeletons, *ASME J. Dyn. Syst. Meas. Contr.* **128**, 14–25 (2006)
- 33.39 H. Kazerooni, L. Huang, J.L. Racine, R. Steger: On the Control of Berkeley Lower Extremity Exoskeleton (BLEEX), *Proc. of IEEE International Conference on Robotics and Automation* (Barcelona, 2005)
- 33.40 S. Kim, G. Anwar, H. Kazerooni: High-speed Communication Network for Controls with Application on the Exoskeleton, *American Control Conference* (Boston, 2004)
- 33.41 S. Kim, H. Kazerooni: High Speed Ring-based distributed Networked control system For Real-Time Multivariable Applications, *ASME International Mechanical Engineering Congress* (Anaheim, 2004)
- 33.42 C. Kirtley: CGA Normative Gait Database, Hong Kong Polytechnic University. (available <http://guardian.curtin.edu.au/cga/data/>)
- 33.43 A. Winter: International Society of Biomechanics, Biomechanical Data Resources, Gait Data. (available <http://www.isbweb.org/data/>)
- 33.44 J. Linsell: CGA Normative Gait Database, Limb Fitting Centre, Dundee, Scotland, Young Adult. (available <http://guardian.curtin.edu.au/cga/data/>)
- 33.45 A. Chu, H. Kazerooni, A. Zoss: On the Biomimetic Design of the Berkeley Lower Extremity Exoskeleton (BLEEX), *Proc. of IEEE International Conference on Robotics and Automation* (Barcelona, 2005)
- 33.46 A. Zoss, H. Kazerooni, A. Chu: On the biomechanical design of the Berkeley lower extremity exoskeleton (BLEEX), *IEEE/ASME Trans. Mechatron.* **11**(2), 128–138 (2006)
- 33.47 T. McGee, J. Raade, H. Kazerooni: Monopropellant-driven free piston hydraulic pump for mobile robotic systems, *J. Dyn. Syst. Meas. Contr.* **126**, 75–81 (2004)
- 33.48 J. Raade, H. Kazerooni, T. McGee: Analysis and design of a novel power supply for mobile robots, *IEEE Trans. Autom. Sci. Eng.* **2**(3), 226–232 (2005)
- 33.49 K. Amundson, J. Raade, N. Harding, H. Kazerooni: Hybrid Hydraulic-Electric Power Unit for Field and Service Robots, *Proc. of IEEE Intelligent Robots and Systems* (Edmonton, 2005)
- 33.50 D.P. Ferris, V. Czerniecki, B. Hannaford: An ankle-foot orthosis powered by artificial muscles, *J. Appl. Biomech.* **21**, 189–197 (2005)
- 33.51 D.P. Ferris, K.E. Gordon, G.S. Sawicki, A. Peethambaran: An improved powered ankle-foot orthosis using proportional myoelectric control, *Gait Posture* **23**, 425–428 (2006)

Mobile and Distributed Robotics

Part E

Part E Mobile and Distributed Robotics

Ed. by Raja Chatila

34 Motion Control of Wheeled Mobile Robots

Pascal Morin, Sophia-Antipolis, France
Claude Samson, Sophia-Antipolis, France

35 Motion Planning and Obstacle Avoidance

Javier Minguez, Zaragoza, Spain
Florent Lamiraux, Toulouse, France
Jean-Paul Laumond, Toulouse, France

36 World Modeling

Wolfram Burgard, Freiburg, Germany
Martial Hebert, Pittsburgh, USA

37 Simultaneous Localization and Mapping

Sebastian Thrun, Stanford, USA
John J. Leonard, Cambridge, USA

38 Behavior-Based Systems

Maja J. Mataríć, Los Angeles, USA
François Michaud, Sherbrooke, Canada

39 Distributed and Cellular Robots

Zack Butler, Rochester, USA
Alfred Rizzi, Waltham, USA

40 Multiple Mobile Robot Systems

Lynne E. Parker, Knoxville, USA

41 Networked Robots

Vijay Kumar, Philadelphia, USA
Daniela Rus, Cambridge, USA
Gaurav S. Sukhatme, Los Angeles, USA

Part E. Mobile Robotics started as a research domain in its own right in the late 1960s with the *Shakey* project at SRI. The seminal paper by N.J. Nilsson "A Mobile Automaton: An Application of Artificial Intelligence Techniques" at IJCAI 1969, already addressed perception, mapping, motion planning, and the notion of control architecture. Those issues would indeed be at the core of mobile robotics research for the following decades. The 1980s boomed with mobile robot projects, and as soon as it was necessary to cope with the reality of the physical world, problems appeared that fostered novel research directions, actually moving away from the original concept in which the robot was just an application of AI techniques. This part addresses all the issues that, put together, are necessary to build and control a mobile robot, except for the mechanical design itself.

Sensing as such is the subject of Part C but the use of sensing for environment mapping and robot localization, central to mobile robot navigation, are developed in this part in Chaps. 36 and 37, with very close links to estimation theory (Part A, Chap. 4). Several issues presented in the foundations (Part A) are revisited in this part and placed in the context and specificities of mobile robots, such as motion control (Chap. 6, Part A and Chap. 34 in this part). Motion planning is addressed in Part A, Chap. 5, whereas the specific problems of motion planning and obstacle avoidance are the subject of Chap. 35. A mobile robot is a system integrating sensing, decision making, and action. Therefore control architecture issues were very much developed in robotics through mobile robotics, often raising controversy. Chapter 38, focusing on behavior-based systems, should probably be read bearing in mind Chap. 8 (Part A) on robotic systems architectures and programming, and Chap. 9 (Part A) on AI reasoning and learning methods for robotics.

What happens when several mobile robots meet together? How does collective behavior emerge and how can it be controlled? These questions arose very early. Answers are developed in Chaps. 39, 40, and 41.

With the development of mobile robots, control issues became a major concern when it appeared necessary to achieve smooth and efficient motion and to follow determined trajectories. In addition, the design of the locomotion structure of wheeled robots was often similar to that of a car, i. e., nonholonomic. Kinematics and control, which were essentially studied in manipulators, entered the field of mobile robotics. The techniques for steering control using mainly feedback control and linear control theory are presented in Chapter 34.

Chapter 35 addresses motion planning from the viewpoint of mobile robotics. In this context, non-holonomic kinematic constraints appeared as a major difficulty and differential geometry as a major tool for achieving the motions computed by classical motion planners. Another difference from classical motion planning addressed here is negotiating unknown or mobile obstacles in uncertain environments, which led to the development of approaches using sensor-based motion.

Mobile robot navigation requires building maps of the environment. The first question addressed in Chapter 36 is therefore: what representations are the most adequate, considering for example that on uneven terrain, the very notion of *obstacle* has a different meaning than indoors? And another question is: how should sensing uncertainties be processed? Furthermore, mobile robots need to map their environment incrementally, while moving in it. Hence partial representations, built from different positions, need to be fused together to construct a consistent map. This requires the robot to know those positions, which are only defined with respect to the environment map itself. Hence localization and mapping must be achieved simultaneously, and Chapter 37 overviews the techniques for solving this SLAM problem.

The mid 1980s was a very rich period for mobile robotics. It is then that three main topics were first addressed: SLAM, nonholonomy, and control architectures. Probably one of the most exciting controversies in the field was raised by Rodney Brooks' subsumption architecture which, by eliminating deliberative components from the architecture, became the paradigm for behavior-based robots. Chapter 38 discusses those architectural concepts.

This part includes three chapters on distributed robotics. Interaction among multiple mobile robots for achieving a common mission or for coordination, through direct or indirect communication, or even no communication at all, robot systems composed of a few machines or large swarms with emerging behaviors, and the underlying algorithms are described in Chapter 40. In that chapter, each robot has its own capacity and individual behavior. Can we design robots from other robots? Can robots reconfigure their own body? Can robots replicate? Chapter 39 overviews modular robots, i. e., robots built from interchangeable parts, that can reconfigure themselves according to the task, to the motion or situation.

Finally, with the development of miniaturization and wireless communication, the notion of distributed robots itself has taken another dimension. Robots can

be scattered everywhere and become pervasive. These new problems are the subject of **Chapter 41**, which concludes this part.

As soon as technology enabled the design at an affordable cost of robots integrating sensing decision and

motion and as research results became available, potential applications in all domains became a serious investigation issue in numerous projects, as will be described in Part F.

34. Motion Control of Wheeled Mobile Robots

Pascal Morin, Claude Samson

This chapter may be seen as a follow up to Chap. 17, devoted to the classification and modeling of basic wheeled mobile robot (WMR) structures, and a natural complement to Chap. 35, which surveys motion planning methods for WMRs. A typical output of these methods is a feasible (or admissible) reference state trajectory for a given mobile robot, and a question which then arises is how to make the physical mobile robot track this reference trajectory via the control of the actuators with which the vehicle is equipped. The object of the present chapter is to bring elements of the answer to this question based on simple and effective control strategies. A first approach would consist in applying *open-loop* steering control laws like those developed in Chap. 35. However, it is well known that this type of control is not robust to modeling errors (the sources of which are numerous) and that it cannot guarantee that the mobile robot will move along the desired trajectory as planned. This is why the methods here presented are based on *feedback* control. Their implementation supposes that one is able to measure the variables involved in the control loop (typically the position and orientation of the mobile robot with respect to either a fixed frame or a path that the vehicle should follow). Throughout this chapter we will assume that these measurements are available continuously in time and that they are not corrupted by noise. In a general manner, robustness considerations will not be discussed in detail, one reason being that, beyond imposed space limitations, a large part of the presented approaches are based on linear control theory. The feedback control laws then inherit the strong robustness properties associated with stable linear systems.

34.1	Background	800
34.1.1	Path Following	800
34.1.2	Stabilization of Trajectories	800
34.1.3	Stabilization of Fixed Postures	801
34.2	Control Models	801
34.2.1	Kinematics Versus Dynamics	801
34.2.2	Modeling in a Frénet Frame	803
34.3	Adaptation of Control Methods for Holonomic Systems	804
34.3.1	Stabilization of Trajectories for a Nonconstrained Point	804
34.3.2	Path Following with No Orientation Control	805
34.4	Methods Specific to Nonholonomic Systems	806
34.4.1	Transformation of Kinematic Models into the Chained Form	807
34.4.2	Tracking of a Reference Vehicle with the Same Kinematics	808
34.4.3	Path Following with Orientation Control	811
34.4.4	Asymptotic Stabilization of Fixed Postures	813
34.4.5	Limitations Inherent to the Control of Nonholonomic Systems	818
34.4.6	Practical Stabilization of Arbitrary Trajectories Based on the Transverse Function Approach	820
34.5	Complementary Issues and Bibliographical Guide	823
	References	825

Results can also be subsequently refined by using complementary, eventually more elaborate, automatic control techniques.

34.1 Background

The control of wheeled mobile robots has been, and still is, the subject of numerous research studies. In particular, nonholonomy constraints associated with these systems have motivated the development of highly nonlinear control techniques. These approaches are addressed in the present chapter, but their exposition is deliberately limited in order to give priority to more classical techniques whose bases, both practical and theoretical, are better established.

For the sake of simplicity, the control methods are developed mainly for unicycle-type and car-like mobile robots, which correspond respectively to the types (2, 0) and (1, 1) in the classification proposed in Chap. 17. Most of the results can in fact be extended/adapted to other mobile robots, in particular to systems with trailers. We will mention the cases where such extensions are straightforward. All reported simulation results, illustrating various control problems and solutions, are carried out for a car-like vehicle, whose kinematics is slightly more complex than that of unicycle-type vehicles.

Recall (Fig. 34.1) that:

1. A *unicycle-type mobile robot* is schematically composed of two independent actuated wheels on a common axle whose direction is rigidly linked to the robot chassis, and one or several passively orientable – or caster – wheels, which are not controlled and serve for sustentation purposes.
2. A (rear-drive) *car-like mobile robot* is composed of a motorized wheeled axle at the rear of the chassis, and one (or a pair of) orientable front steering wheel(s).

Note also, as illustrated by the diagram below (Fig. 34.2), that a car-like mobile robot can be viewed (at least kinematically) as a unicycle-type mobile robot to which a trailer is attached.

Three generic control problems are studied in this chapter, as described below.

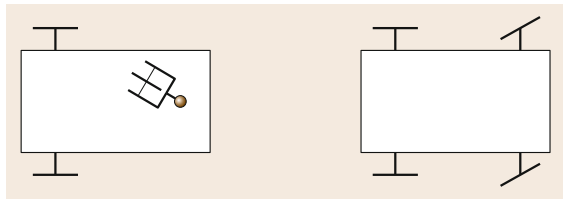


Fig. 34.1 Unicycle-type (left) and car-like (right) mobile robots

34.1.1 Path Following

Given a curve \mathcal{C} on the plane, a (nonzero) longitudinal velocity v_0 for the robot chassis, and a point P attached to the chassis, the goal is to have the point P follow the curve \mathcal{C} when the robot moves with the velocity v_0 . The variable that one has to stabilize at zero is thus the distance between the point P and the curve (i. e., the distance between P and the closest point M on \mathcal{C}). This type of problem typically corresponds to driving on a road while trying to maintain the distance between the vehicle chassis and the side of the road constant. Automatic wall following is another possible application.

34.1.2 Stabilization of Trajectories

This problem differs from the previous one in that the vehicle's longitudinal velocity is no longer predetermined because one also aims to monitor the distance gone along the curve \mathcal{C} . This objective supposes that the geometric curve \mathcal{C} is complemented with a time schedule, i. e., that it is parameterized with the time variable t . This boils down to defining a trajectory $t \mapsto (x_r(t), y_r(t))$ with respect to a reference frame \mathcal{F}_0 . Then the goal is to stabilize the position error vector $(x(t) - x_r(t), y(t) - y_r(t))$ at zero, with $(x(t), y(t))$ denoting the coordinates of point P in \mathcal{F}_0 at time t . The problem may also be formulated as one of controlling the vehicle in order to track a reference vehicle whose trajectory is given by $t \mapsto (x_r(t), y_r(t))$. Note that perfect tracking is achievable only if the reference trajectory is feasible for the physical vehicle, and that a trajectory which is feasible for a unicycle-type vehicle is not necessarily feasible for a car-like vehicle. Also, in addition to monitoring the position $(x(t), y(t))$ of the robot, one may be willing to control the chassis orientation $\theta(t)$ at a desired reference value $\theta_r(t)$ associated with the orientation of the reference vehicle. For a nonholonomic unicycle-type robot, a reference trajectory $(x_r(t), y_r(t), \theta_r(t))$ is feasible if it is produced by a reference vehicle which has the

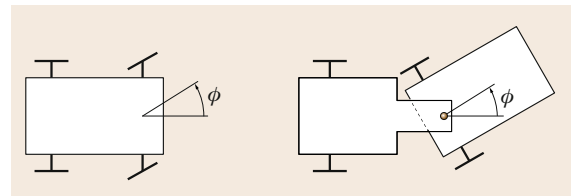


Fig. 34.2 Analogy car/unicycle with trailer

same kinematic limitations as the physical robot. For instance, most trajectories produced by an omnidirectional vehicle (omnibible vehicle in the terminology of Chap. 17) are not feasible for a nonholonomic mobile robot. However, nonfeasibility does not imply that the reference trajectory cannot be tracked in an approximate manner, i. e., with small (although nonzero) tracking errors. This justifies the introduction of the concept of *practical* stabilization, as opposed to *asymptotic* stabilization when the tracking errors converge to zero. The last part of this chapter will be devoted to a recent, and still prospective, control approach for the practical stabilization of trajectories which are not necessarily feasible.

34.1.3 Stabilization of Fixed Postures

Let \mathcal{F}_1 denote a frame attached to the robot chassis. In this chapter, we call a robot posture (or situation) the association of the position of a point P located on the robot chassis with the orientation $\theta(t)$ of \mathcal{F}_1 with respect to a fixed frame \mathcal{F}_0 in the plane of motion. For this last problem, the objective is to stabilize at zero the posture vector $\xi(t) = (x(t), y(t), \theta(t))$, with $(x(t), y(t))$ denoting the position of P expressed in \mathcal{F}_0 . Although a fixed desired (or reference) posture is obviously a particular case of a feasible trajectory, this problem cannot be solved by classical control methods.

34.2 Control Models

34.2.1 Kinematics Versus Dynamics

Relation (17.29) in Chap. 17 provides a general configuration dynamic model for WMRs. Its particularization to the case of unicycle-type and car-like mobile robots gives

$$H(q)\dot{u} + F(q, u)u = \Gamma(\phi)\tau \quad (34.1)$$

with q denoting a robot's configuration vector, u a vector of independent velocity variables associated with the robot's degrees of freedom, $H(q)$ a reduced inertia matrix (which is invertible for any q), $F(q, u)u$ a vector of forces combining the contribution of Coriolis and wheel-ground contact forces, ϕ the orientation angle of the car's steering wheel, Γ an invertible control matrix (which is constant in the case of a unicycle-type vehicle), and τ a vector of independent motor torques (whose dimension is equal to the number of degrees of freedom in the case of full actuation, i. e., equal to two for the

The chapter is organized as follows. Section 34.2 is devoted to the choice of control models and the determination of modeling equations associated with the path-following control problem. In Sect. 34.3, the problems of path following and trajectory stabilization in position are studied under an assumption upon the location of the point P chosen on the robot chassis. This assumption implies that the motion of this point is not constrained. It greatly simplifies the resolution of the considered problems. However, a counterpart of this simplification is that the stability of the robot's orientation is not always guaranteed, in particular during phases when the sign of the robot's longitudinal velocity is not constant. The assumption upon P is removed in Sect. 34.4, and both problems are reconsidered, together with the problem of stabilizing a fixed posture. At the end of this section, a certain number of shortcomings and limitations inherent to the objective of asymptotic stabilization are pointed out. They can be circumvented by considering an objective of practical stabilization instead. Some elements of a recent, and still prospective, control approach developed with this point of view – based on the use of so-called *transverse functions* – are presented in the Sect. 34.4.6. Finally, a few complementary issues on the feedback control of mobile robots are briefly discussed in the concluding Sect. 34.5, with a list of commented references for further reading on WMRs motion control.

vehicles considered herein). In the case of a unicycle-type vehicle, a configuration vector is composed of the components of the chassis posture vector ξ and the orientation angles of the castor wheels (with respect to the chassis). In the case of a car-like vehicle, a configuration vector is composed of the components of ξ and the steering wheel angle ϕ .

To be complete, this dynamic model must be complemented with kinematic equations in the form (the relation (17.30))

$$\dot{q} = S(q)u \quad (34.2)$$

from which one can extract a reduced kinematic model (the relation (17.33))

$$\dot{z} = B(z)u \quad (34.3)$$

with $z = \xi$, in the case of a unicycle-type vehicle, and $z = (\xi, \phi)$ in the case of a car-like vehicle.

In the automatic control terminology, the complete dynamic model (34.1–34.2) forms a *control system* which can be written as $\dot{X} = f(X, \tau)$ with $X = (q, u)$ denoting the state vector of this system, and τ the vector of control inputs. The kinematic models (34.2) and (34.3) are also control systems with respective state vectors q and z and control vector u . Any of these models can be used for control design and analysis purposes. In the remainder of this chapter, we have chosen to work with the kinematic model (34.3). By analogy with the motion control of manipulator arms, this boils down to using a model with velocity control inputs, rather than a model with torque control inputs. The main reasons for this choice are the following:

1. The kinematic model is simpler than the dynamic one. In particular, it does not involve a certain number of matrix-valued functions whose precise determination relies on the knowledge of numerous parameters associated with the vehicle and its actuators (geometric repartition of constitutive bodies, masses and mass moments of inertia, coefficients of reduction in the transmission of torques produced by the motors, etc.). For many applications, it is not necessary to know all these terms precisely.
2. In the case of robots actuated with electrical motors, these motors are frequently supplied with *low-level* velocity control loops which take a desired angular velocity as input and stabilize the motor angular velocity at this value. If the regulation loop is efficient, the difference between the desired and actual velocities remains small, even when the desired velocity and the motor load vary continuously (at least within a certain range). This type of robustness allows in turn to view the desired velocity as a free control variable. Many controllers supplied with industrial manipulator arms are based on this principle.

3. If the servo loops evoked above, whose role is to decouple the kinematics from the dynamics of the vehicle, are not present, one can design them and even improve their performance by using the information that one has of the terms involved in the dynamic equation (34.1). For instance, assume that the torques produced by the actuators can be used as control inputs, a simple way to proceed (at least theoretically) consists in applying the so-called *computed torque* method. The idea is to linearize the dynamic equation by setting

$$\tau = \Gamma(\phi)^{-1} [H(q)w + F(q, u)u].$$

This yields the simple decoupled linear control system $\dot{u} = w$ with the variable w , homogeneous to a vector of accelerations, playing the role of a new control input vector. This latter equation indicates that the problem of controlling the vehicle with motor torques can be brought back to a problem with acceleration control inputs. It is usually not difficult to deduce a control solution to this problem from a velocity control solution devised by using a kinematic model. For instance,

$$w = -k(u - u^*(z, t)) + \frac{\partial u^*}{\partial z}(z, t)B(z)u + \frac{\partial u^*}{\partial t}(z, t)$$

with $k > 0$ is a solution if u^* is a differentiable kinematic solution and

$$u = u^*(z, t) + (u(0) - u^*(z_0, 0))e^{-kt}$$

is also a solution.

For the unicycle-type mobile robot, the kinematic model (34.3) used from now on is

$$\begin{cases} \dot{x} = u_1 \cos \theta, \\ \dot{y} = u_1 \sin \theta, \\ \dot{\theta} = u_2, \end{cases} \quad (34.4)$$

where (x, y) represents the coordinates of the point P_m located at mid-distance of the actuated wheels, and the angle θ characterizes the robot's chassis orientation (Fig. 34.3). In this equation, u_1 represents the intensity of the vehicle's longitudinal velocity, and u_2 is the chassis instantaneous velocity of rotation. The variables u_1 and u_2 are themselves related to the angular velocity of the actuated wheels via the one-to-one relations

$$\begin{aligned} u_1 &= \frac{r}{2}(\dot{\psi}_r + \dot{\psi}_\ell) \\ u_2 &= \frac{r}{2R}(\dot{\psi}_r - \dot{\psi}_\ell) \end{aligned}$$

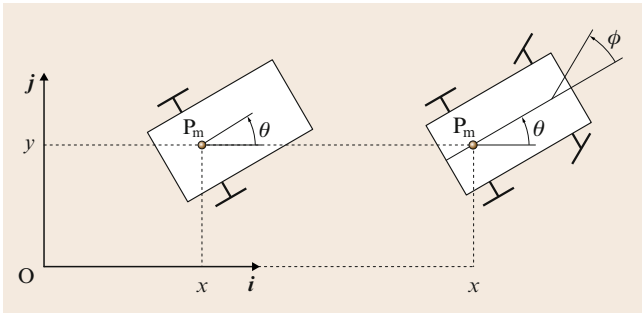


Fig. 34.3 Configuration variables

with r the wheels' radius, R the distance between the two actuated wheels, and $\dot{\psi}_r$ (resp. $\dot{\psi}_\ell$) the angular velocity of the right (resp. left) rear wheel.

For the car-like mobile robot, the kinematic model (34.3) used from now on is

$$\begin{cases} \dot{x} = u_1 \cos \theta, \\ \dot{y} = u_1 \sin \theta, \\ \dot{\theta} = \frac{u_1}{L} \tan \phi, \\ \dot{\phi} = u_2, \end{cases} \quad (34.5)$$

where ϕ represents the vehicle's steering wheel angle, and L is the distance between the rear and front wheels' axes. In all forthcoming simulations, L is set equal to 1.2 m.

34.2.2 Modeling in a Frénet Frame

The object of this subsection is to generalize the previous kinematic equations when the reference frame is a Frénet frame. This generalization will be used later on when addressing the path following problem.

Let us consider a curve \mathcal{C} in the plane of motion, as illustrated on Fig. 34.4, and let us define three frames \mathcal{F}_0 , \mathcal{F}_m , and \mathcal{F}_s , as follows. $\mathcal{F}_0 = \{0, i, j\}$ is a fixed frame, $\mathcal{F}_m = \{P_m, i_m, j_m\}$ is a frame attached to the mobile robot with its origin – the point P_m – located on the rear wheels axle, at the mid-distance of the wheels, and $\mathcal{F}_s = \{P_s, i_s, j_s\}$, which is indexed by the curve's curvilinear abscissa s , is such that the unit vector i_s tangents \mathcal{C} .

Consider now a point P attached to the robot chassis, and let (l_1, l_2) denote the coordinates of P expressed in the basis of \mathcal{F}_m . To determine the equations of motion of P with respect to the curve \mathcal{C} let us introduce three variables s , d , and θ_e , defined as follows.

- s is the curvilinear abscissa at the point P_s obtained by projecting P orthogonally on \mathcal{C} . This point exists and is unique if the point P is close enough to the curve. More precisely, it suffices that the distance between P and the curve be smaller than the lower bound of the curve radii. We will assume that this condition is satisfied.
- d is the ordinate of P in the frame \mathcal{F}_s ; its absolute value is also the distance between P and the curve.
- $\theta_e = \theta - \theta_s$ is the angle characterizing the orientation of the robot chassis with respect to the frame \mathcal{F}_s .

Let us now determine \dot{s} , \dot{d} , and $\dot{\theta}_e$. By definition of the curvature $c(s)$ of \mathcal{C} at P_s , i. e., $c(s) = \partial\theta_s/\partial s$, one

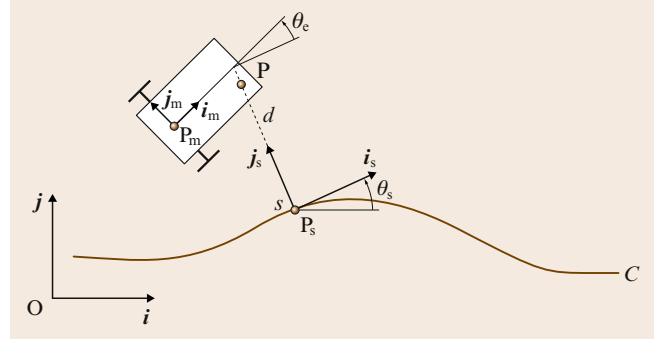


Fig. 34.4 Representation in a Frénet frame

deduces from (34.4) that

$$\dot{\theta}_e = u_2 - \dot{s}c(s). \quad (34.6)$$

Since $P_s P = d j_s$, by using the equality $d \mathbf{O}P_s / dt = \dot{s} i_s$ it first emerges that

$$\begin{aligned} \frac{\partial \mathbf{O}P}{\partial t} &= \frac{\partial \mathbf{O}P_s}{\partial t} + \dot{d} j_s - dc(s) \dot{s} i_s \\ &= \dot{s}(1 - dc(s)) i_s + \dot{d} j_s. \end{aligned} \quad (34.7)$$

One also has $P_m P = l_1 i_m + l_2 j_m$. Since $d \mathbf{O}P_m / dt = u_1 i_m$, one gets

$$\begin{aligned} \frac{\partial \mathbf{O}P}{\partial t} &= \frac{\partial \mathbf{O}P_m}{\partial t} + l_1 u_2 j_m - l_2 u_2 i_m \\ &= (u_1 - l_2 u_2) i_m + l_1 u_2 j_m \\ &= (u_1 - l_2 u_2)(\cos \theta_e i_s + \sin \theta_e j_s) \\ &\quad + l_1 u_2 (-\sin \theta_e i_s + \cos \theta_e j_s) \\ &= [(u_1 - l_2 u_2) \cos \theta_e - l_1 u_2 \sin \theta_e] i_s \\ &\quad + [(u_1 - l_2 u_2) \sin \theta_e + l_1 u_2 \cos \theta_e] j_s. \end{aligned} \quad (34.8)$$

By forming the scalar products of the vectors in (34.7) and (34.8) with i_s and j_s , and by using (34.6), one finally obtains the following system of equations:

$$\begin{cases} \dot{s} = \frac{1}{1 - dc(s)} [(u_1 - l_2 u_2) \cos \theta_e - l_1 u_2 \sin \theta_e], \\ \dot{d} = (u_1 - l_2 u_2) \sin \theta_e + l_1 u_2 \cos \theta_e, \\ \dot{\theta}_e = u_2 - \dot{s}c(s). \end{cases} \quad (34.9)$$

These equations are a generalization of (34.4). To verify this, it suffices to take P as the origin of the frame \mathcal{F}_m (i. e., $l_1 = l_2 = 0$) and identify the axis (O, i) of the frame \mathcal{F}_0 with the curve \mathcal{C} . Then $s = x$, $c(s) = 0$ ($\forall s$), and, by setting $y \equiv d$ and $\theta \equiv \theta_e$, one recovers (34.4) exactly.

For car-like vehicles, one easily verifies, by using (34.5), that the system (34.9) becomes

$$\begin{cases} \dot{s} = \frac{u_1}{1-dc(s)} \left[\cos \theta_e - \frac{\tan \phi}{L} (l_2 \cos \theta_e + l_1 \sin \theta_e) \right], \\ \dot{d} = u_1 \left[\sin \theta_e + \frac{\tan \phi}{L} (l_1 \cos \theta_e - l_2 \sin \theta_e) \right], \\ \dot{\theta}_e = \frac{u_1}{L} \tan \phi - \dot{c}(s), \\ \dot{\phi} = u_2, \end{cases} \quad (34.10)$$

(it suffices to replace $u_2 = \dot{\theta}$ in (34.9) by the new value of $\dot{\theta}$: $(u_1 \tan \phi)/L$). To summarize, we have shown the following result.

Proposition 34.1

The kinematic equations of unicycle-type and car-like vehicles, expressed with respect to a Frénet frame, are given by the systems (34.9) and (34.10), respectively.

34.3 Adaptation of Control Methods for Holonomic Systems

We address in this section the problems of trajectory stabilization and path following. When we defined these problems in the introduction, we considered a reference point P attached to the robot chassis. It turns out that the choice of this point is important. Indeed, consider for instance the equations (34.9) for a unicycle point P when \mathcal{C} is the axis (O, i) . Then, $s = x_P$, $d = y_P$, and $\theta_e = \theta$ represent the robot's posture with respect to the fixed reference frame \mathcal{F}_0 . There are two possible cases depending on whether P is, or is not, located on the actuated wheels axle. Let us consider the first case, for which $l_1 = 0$. From the first two equations of (34.9), one has

$$\dot{x}_P = (u_1 - l_2 u_2) \cos \theta, \quad \dot{y}_P = (u_1 - l_2 u_2) \sin \theta.$$

These relations indicate that P can move only in the direction of the vector $(\cos \theta, \sin \theta)$. This is a direct consequence of the nonholonomy constraint to which the vehicle is subjected. Now, if P is not located on the wheels axle, then

$$\begin{pmatrix} \dot{x}_P \\ \dot{y}_P \end{pmatrix} = \begin{pmatrix} \cos \theta & -l_1 \sin \theta \\ \sin \theta & l_1 \cos \theta \end{pmatrix} \begin{pmatrix} 1 & -l_2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}. \quad (34.11)$$

The fact that the two square matrices in the right-hand side of this equality are invertible indicates that \dot{x}_P and \dot{y}_P can take any values, and thus that the motion of P is not constrained. By analogy with holonomic manipulator arms, this means that P may be seen as the extremity of a two-degree-of-freedom (2-DOF) manipulator, and thus that it can be controlled by applying the same control laws as those used for manipulators. In this section, we assume that the point P, used to characterize the robot's position, is chosen away from the rear wheels axle. In this case we will see that the problems of trajectory stabilization and path following can be solved very

simply. However, as shown in the subsequent section, choosing P on the wheels axle may also be of interest in order to better control the vehicle's orientation.

34.3.1 Stabilization of Trajectories for a Nonconstrained Point

Unicycle

Consider a differentiable reference trajectory $t \mapsto (x_r(t), y_r(t))$ in the plane. Let $e = (x_P - x_r, y_P - y_r)$ denote the tracking error in position. The control objective is to asymptotically stabilize this error at zero. In view of (34.11), the error equations are

$$\dot{e} = \begin{pmatrix} \cos \theta & -l_1 \sin \theta \\ \sin \theta & l_1 \cos \theta \end{pmatrix} \begin{pmatrix} u_1 - l_2 u_2 \\ u_2 \end{pmatrix} - \begin{pmatrix} \dot{x}_r \\ \dot{y}_r \end{pmatrix}. \quad (34.12)$$

Introducing new control variables (v_1, v_2) defined by

$$\begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} \cos \theta & -l_1 \sin \theta \\ \sin \theta & l_1 \cos \theta \end{pmatrix} \begin{pmatrix} u_1 - l_2 u_2 \\ u_2 \end{pmatrix} \quad (34.13)$$

the equations (34.12) become simply

$$\dot{e} = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} - \begin{pmatrix} \dot{x}_r \\ \dot{y}_r \end{pmatrix}.$$

The classical techniques of stabilization for linear systems can then be used. For instance, one may consider a proportional feedback control with precompensation such as

$$\begin{aligned} v_1 &= \dot{x}_r - k_1 e_1 = \dot{x}_r - k_1 (x_P - x_r), & (k_1 > 0), \\ v_2 &= \dot{y}_r - k_2 e_2 = \dot{y}_r - k_2 (y_P - y_r), & (k_2 > 0), \end{aligned}$$

which yields the closed-loop equation $\dot{e} = -Ke$. Of course, this control can be rewritten for the initial control variables u , since the mapping $(u_1, u_2) \mapsto (v_1, v_2)$ is bijective.

Unicycle with Trailers

The previous technique extends directly to the case of a unicycle-type vehicle to which one or several trailers are hooked, provided that the point P is chosen away from the actuated wheels axle and on the side opposite to the trailers. Besides, it is preferable in practice that the robot's longitudinal velocity u_1 remains positive all the time in order to prevent the relative orientations between all vehicles (i.e., the non-actively controlled variables involved in the system's *zero dynamics*) to take overly large values (the jackknife effect). This issue will be discussed further in Sect. 34.4.

Car

This technique also extends to car-like vehicles by choosing a point P attached to the steering wheel frame and not located on the steering wheel axle.

34.3.2 Path Following with No Orientation Control

Unicycle

Let us adopt the notation of Fig. 34.4 to address the problem of following a path associated with a curve \mathcal{C} in the plane. The control objective is to stabilize the distance d at zero. From (34.9), one has

$$\dot{d} = u_1 \sin \theta_e + u_2(-l_2 \sin \theta_e + l_1 \cos \theta_e). \quad (34.14)$$

Recall that in this case the vehicle's longitudinal velocity u_1 is either imposed or prespecified. We will assume that the product $l_1 u_1$ is positive, i.e., the position of the point P with respect to the actuated wheels axle is chosen in relation to the sign of u_1 . This assumption will be removed in Sect. 34.4. To simplify, we will also assume that $l_2 = 0$, i.e., the point P is located on the axis (P_m, i_m). Let us then consider the following feedback control law

$$u_2 = -\frac{u_1}{l_1 \cos \theta_e} \sin \theta_e - \frac{u_1}{\cos \theta_e} k(d, \theta_e) d \quad (34.15)$$

with k a continuous, strictly positive, function on $\mathbb{R} \times (-\pi/2, \pi/2)$ such that $k(d, \pm\pi/2) = 0$. Since $l_2 = 0$, applying the control (34.15) to (34.14) gives

$$\dot{d} = -l_1 u_1 k(d, \theta_e) d.$$

Since $l_1 u_1$ and k are strictly positive, this relation implies that $|d|$ is nonincreasing along any trajectory of the controlled system. For convergence of d to zero, it suffices that

1. the sign of u_1 remains the same,

2. $\pi/2 - |\theta_e(t)| > \epsilon > 0$ for all t , and
- 3.

$$\int_0^t |u_1(s)| ds \rightarrow +\infty \quad \text{when } t \rightarrow +\infty.$$

This latter condition is satisfied, for instance, when u_1 is constant. In this case, d converges to zero exponentially. There just remains to examine the conditions under which u_2 , as given by (34.15), is always defined. Since the function in the right-hand side of (34.15) is not defined when $\cos \theta_e = 0$, we are going to determine conditions on the system parameters and the initial state values the satisfaction of which implies that $\cos \theta_e$ cannot approach zero. To this purpose, let us consider the limit value of $\dot{\theta}_e$ when θ_e tends to $\pi/2$ (resp. $-\pi/2$) from below (resp. from above). By using (34.9), (34.15), and the fact that $l_2 = 0$, a simple calculation shows that

$$\begin{aligned} \dot{\theta}_e = u_1 & \left[-\frac{c(s)}{1 - dc(s)} \cos \theta_e \right. \\ & - \left(1 + \frac{l_1 c(s)}{1 - dc(s)} \sin \theta_e \right) \\ & \times \left(\frac{\tan \theta_e}{l_1} + \frac{k(d, \theta_e) d}{\cos \theta_e} \right) \Big]. \end{aligned}$$

Let us assume first that θ_e tends to $\pi/2$ from below. Then the sign of $\dot{\theta}_e$ is given, in the limit, by the sign of

$$-u_1 \left(1 + \frac{l_1 c(s)}{1 - dc(s)} \right) \frac{1}{l_1}.$$

To prevent θ_e from reaching $\pi/2$ it suffices that this sign be negative. It is so if

$$\left| \frac{l_1 c(s)}{1 - dc(s)} \right| < 1. \quad (34.16)$$

Now, if θ_e tends to $-\pi/2$ from above, the sign of $\dot{\theta}_e$ is given, in the limit, by the sign of

$$-u_1 \left(1 - \frac{l_1 c(s)}{1 - dc(s)} \right) \left(\frac{-1}{l_1} \right).$$

To prevent θ_e from reaching $-\pi/2$ it suffices that this sign be positive, and such is the case if (34.16) is true. From this analysis one obtains the following proposition.

Proposition 34.2

Consider the path following problem for a unicycle-type mobile robot with

- [A.] a strictly positive, or strictly negative, longitudinal velocity u_1 .
- [B.] a reference point P of coordinates $(l_1, 0)$ in the vehicle's chassis frame, with $l_1 u_1 > 0$.

Let k denote a continuous function, strictly positive on $\mathbb{R} \times (-\pi/2, \pi/2)$, and such that $k(d, \pm\pi/2) = 0$ for every d (for instance, $k(d, \theta_e) = k_0 \cos \theta_e$). Then, for any initial conditions $(s(0), d(0), \theta_e(0))$ such that

$$\theta_e(0) \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right), \quad \frac{l_1 c_{\max}}{1 - |d(0)| c_{\max}} < 1$$

with $c_{\max} = \max_s |c(s)|$, the feedback control

$$u_2 = -\frac{u_1 \tan \theta_e}{l_1} - u_1 \frac{k(d, \theta_e) d}{\cos \theta_e}$$

makes the distance $|d|$ between P and the curve nonincreasing, and makes it converge to zero if

$$\int_0^t |u_1(s)| ds \rightarrow +\infty \quad \text{when } t \rightarrow +\infty.$$

Unicycle with Trailers

The above result applies also to this system, except that u_1 has to be positive in order to avoid jackknife effects which otherwise may (will, if u_1 is kept negative long enough) occur because the orientation angles between the vehicles are not actively monitored.

Car

This control technique thus also applies to this case by considering a point P attached to the steering wheel frame, with u_1 positive.

34.4 Methods Specific to Nonholonomic Systems

The control technique presented in the previous section has the advantage of being simple. However, it is not well suited for all control purposes. One of its main limitations is that it relies on the invariance of the sign of the robot's longitudinal velocity (see the assumptions in Proposition 34.2). For systems with trailers, this velocity is further required to be positive. This condition/restriction is related to the fact that the orientation variables are not actively monitored. To understand its nature better, let us consider the control solution given in Proposition 34.2 for $u_1 > 0$, and assume that this control is applied with u_1 negative (and constant, for instance), with the point P being unchanged. Figure 34.5 illustrates a possible *scenario*. The chosen curve is a simple straight line and we assume that, at time $t = 0$, P is already on this curve (i. e., $d = 0$). If u_1 is negative, while

P is bound to stay on the line, the magnitude $|\theta_e|$ of the robot's orientation angle increases rapidly. The angle θ_e reaches $-\pi/2$ at $t = 1$. At this time the control expression is no longer defined (explosion in finite time) and, since the velocity vector has become orthogonal to the curve, the point P can no longer stay on the curve.

The explanation for this behavior is as follows. Since the orientation is not controlled, the variable θ_e has its own, a priori unknown, dynamics. It can be stable, or unstable. For the considered control solution, we have shown that it is stable when $u_1 > 0$. In particular, θ_e remains in the domain of definition $(-\pi/2, \pi/2)$ of the feedback law for u_2 . When $u_1 < 0$, this dynamics becomes unstable and θ_e reaches the border of the control law's domain of definition $(-\pi/2, \pi/2)$. This instability upon the part of the system which is not directly controlled (the system's *zero dynamics*, in the control terminology) occurs in the same manner when trailers are hooked to a leading vehicle. When the longitudinal velocity is positive, the vehicle has a *pulling* action which tends to align the followers along the curve. In the other case, the leader has a *pushing* action which tends to misalign them (the jackknife effect). In order to remove this constraint on the sign of the longitudinal velocity, the control has to be designed so that all orientation angles are actively stabilized. An indirect way to do this consists in choosing the point P on the actuated wheels axle, at the mid-distance of the wheels, for instance. In

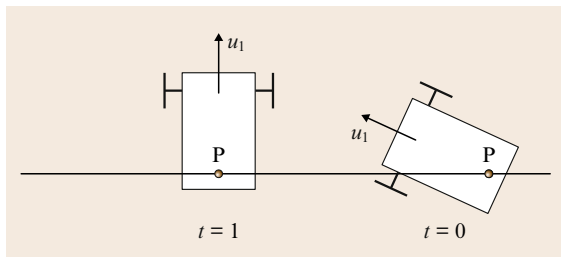


Fig. 34.5 Path-following instability with reverse longitudinal velocity

this case, the nonholonomy constraints intervene much more explicitly, and the control can no longer be obtained by applying the techniques used for holonomic manipulators.

This section is organized as follows. First, the modeling equations with respect to a Frénet frame are recast into a canonical form called the chained form. From there, a solution to the path-following problem with active stabilization of the vehicle's orientation is worked out. The problem of (feasible) trajectory stabilization is also revisited with the complementary objective of controlling the vehicle's orientation. The asymptotic stabilization of fixed postures is then addressed. Finally, some comments on the limitations of the proposed control strategies, in relation to the objective of asymptotic stabilization, serve to motivate and introduce a new control approach developed in the subsequent section.

34.4.1 Transformation of Kinematic Models into the Chained Form

In the next chapter dedicated to path planning, it is shown how the kinematic equations of the mobile robots here considered (unicycle-type, car-like, with trailers) can be transformed into the chain form via a change of state and control variables. In particular, the equations of a unicycle (34.4), and those of a car (34.5), can be transformed into a three-dimensional and a four-dimensional chained system, respectively. Those of a unicycle-type vehicle with N trailers yield a chained system of dimension $N + 3$ when the trailers are hooked to each other in a specific way. As shown below, this transformation can be generalized to the kinematic models derived with respect to a Frénet frame. The result will be given only for the unicycle and car cases (equations (34.9) and (34.10)), but it also holds when trailers are hooked to such vehicles. The reference point P is now chosen at the mid-distance of the vehicle's rear wheels (or at the mid-distance of the wheels of the last trailer, when trailers are involved).

Let us start with the unicycle case. Under the assumption that P corresponds to the origin of \mathcal{F}_m , one has $l_1 = l_2 = 0$ so that the system (34.9) simplifies to

$$\begin{cases} \dot{s} = \frac{u_1}{1-dc(s)} \cos \theta_e, \\ \dot{d} = u_1 \sin \theta_e, \\ \dot{\theta}_e = u_2 - \dot{s}c(s). \end{cases} \quad (34.17)$$

Let us determine a change of coordinates and control variables $(s, d, \theta_e, u_1, u_2) \mapsto (z_1, z_2, z_3, v_1, v_2)$ allowing to (locally) transform (34.17) into the three-

dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1, \\ \dot{z}_2 = v_1 z_3, \\ \dot{z}_3 = v_2. \end{cases} \quad (34.18)$$

By first setting

$$z_1 = s, \quad v_1 = \dot{s} = \frac{u_1}{1-dc(s)} \cos \theta_e,$$

we already obtain $\dot{z}_1 = v_1$. This implies that

$$\begin{aligned} \dot{d} &= u_1 \sin \theta_e = \frac{u_1}{1-dc(s)} \cos \theta_e [1-dc(s)] \tan \theta_e \\ &= v_1 [1-dc(s)] \tan \theta_e. \end{aligned}$$

We then set $z_2 = d$ et $z_3 = [1-dc(s)] \tan \theta_e$, so that the above equation becomes $\dot{z}_2 = v_1 z_3$. Finally, we define

$$\begin{aligned} v_2 &= \dot{z}_3 \\ &= [-\dot{d}c(s) - d \frac{\partial c}{\partial s} \dot{s}] \tan \theta_e \\ &\quad + [1-dc(s)](1+\tan^2 \theta_e) \dot{\theta}_e. \end{aligned}$$

The equations (34.18) are satisfied with the variables z_i and v_i so defined.

From this construction it is simple to verify that the mapping $(s, d, \theta_e) \mapsto z$ is a *local* change of coordinates defined on $\mathbb{R}^2 \times (-\pi/2, \pi/2)$ (to be more rigorous, one should also take the constraint $|d| < 1/c(s)$ into account). Let us finally remark that the change of control variables involves the derivative $(\partial c / \partial s)$ of the path's curvature (whose knowledge is thus needed for the calculations). One can similarly transform the car's equations into a four-dimensional chained system, although the calculations are slightly more cumbersome. Let us summarize these results in the following proposition.

Proposition 34.3

The change of coordinates and of control variables $(s, d, \theta_e, u_1, u_2) \mapsto (z_1, z_2, z_3, v_1, v_2)$ defined by

$$\begin{aligned} (z_1, z_2, z_3) &= (s, d, [1-dc(s)] \tan \theta_e) \\ (v_1, v_2) &= (\dot{z}_1, \dot{z}_3) \end{aligned}$$

transforms the model (34.17) of a unicycle-type vehicle into a three-dimensional chained system.

Similarly, the change of coordinates and control variables $(s, d, \theta_e, \phi, u_1, u_2) \mapsto (z_1, z_2, z_3, z_4, v_1, v_2)$

defined by

$$(z_1, z_2, z_3, z_4) = \left\{ s, d, [1 - dc(s)] \tan \theta_e, \right. \\ \left. -c(s)[1 - dc(s)](1 + 2\tan^2 \theta_e) \right. \\ \left. -d \frac{\partial c}{\partial s} \tan \theta_e \right. \\ \left. + [1 - dc(s)]^2 \frac{\tan \phi}{L} \frac{1 + \tan^2 \theta_e}{\cos \theta_e} \right\}, \\ (v_1, v_2) = (\dot{z}_1, \dot{z}_4)$$

transforms the model (34.10) of a car-like vehicle (with $l_1 = l_2 = 0$) into a four-dimensional chained system.

34.4.2 Tracking of a Reference Vehicle with the Same Kinematics

Let us now consider the problem of tracking, in both position and orientation, a reference vehicle (Fig. 34.6). Contrary to what happens when the control objective is limited to the tracking in position only (Sect. 34.3.1), the choice of the reference point P is of lesser importance because, whatever P, most reference trajectories $t \mapsto (x_r(t), y_r(t), \theta_r(t))$ are not feasible for the state vector (x_P, y_P, θ) . For simplicity, we choose P as the origin P_m of the robot's chassis frame \mathcal{F}_m .

Although the terminology is rather loose, the *tracking problem* is usually associated, in the control literature, with the problem of *asymptotically stabilizing* the reference trajectory. In this case, a necessary condition for the existence of a control solution is that the reference is feasible. Feasible trajectories $t \mapsto (x_r(t), y_r(t), \theta_r(t))$ are smooth time functions which are solution to the robot's kinematic model for some specific control input $t \mapsto \mathbf{u}_r(t) = (u_{1,r}(t), u_{2,r}(t))^T$, called the *reference control*. For a unicycle-type robot for example, this means in view of (34.4) that

$$\begin{cases} \dot{x}_r = u_{1,r} \cos \theta_r, \\ \dot{y}_r = u_{1,r} \sin \theta_r, \\ \dot{\theta}_r = u_{2,r}. \end{cases} \quad (34.19)$$

In other words, feasible reference trajectories correspond to the motion of a reference frame $\mathcal{F}_r = \{P_r, \mathbf{i}_r, \mathbf{j}_r\}$ rigidly attached to a reference unicycle-type robot, with P_r (alike $P = P_m$) located at the mid-distance of the actuated wheels (see Fig. 34.6). From there, the problem is to determine a feedback control which asymptotically stabilizes the tracking error $(x - x_r, y - y_r, \theta - \theta_r)$

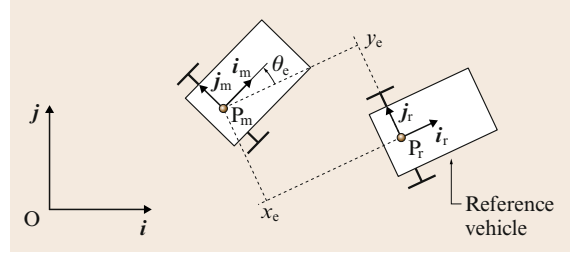


Fig. 34.6 Reference vehicle and error coordinates

at zero, with (x_r, y_r) being the coordinates of P_r in \mathcal{F}_0 , and θ_r the oriented angles between \mathbf{i} and \mathbf{i}_r . One can proceed as in the path-following case, first by establishing the error equations with respect to the frame \mathcal{F}_r , then by transforming these equations in the chain form via a change of variables like the one used to transform the kinematic equations of a mobile robot into a chained system, and finally by designing stabilizing control laws for the transformed system.

Expressing the tracking error in position $(x - x_r, y - y_r)$ with respect to the frame \mathcal{F}_r gives the vector (Fig. 34.6)

$$\begin{pmatrix} x_e \\ y_e \end{pmatrix} = \begin{pmatrix} \cos \theta_r & \sin \theta_r \\ -\sin \theta_r & \cos \theta_r \end{pmatrix} \begin{pmatrix} x - x_r \\ y - y_r \end{pmatrix}. \quad (34.20)$$

Calculating the time derivative of this vector yields

$$\begin{aligned} \begin{pmatrix} \dot{x}_e \\ \dot{y}_e \end{pmatrix} &= \dot{\theta}_r \begin{pmatrix} -\sin \theta_r & \cos \theta_r \\ -\cos \theta_r & -\sin \theta_r \end{pmatrix} \begin{pmatrix} x - x_r \\ y - y_r \end{pmatrix} \\ &\quad + \begin{pmatrix} \cos \theta_r & \sin \theta_r \\ -\sin \theta_r & \cos \theta_r \end{pmatrix} \begin{pmatrix} \dot{x} - \dot{x}_r \\ \dot{y} - \dot{y}_r \end{pmatrix} \\ &= \begin{pmatrix} u_{2,r} y_e + u_1 \cos(\theta - \theta_r) - u_{1,r} \\ -u_{2,r} x_e + u_1 \sin(\theta - \theta_r) \end{pmatrix}. \end{aligned}$$

By denoting $\theta_e = \theta - \theta_r$, the orientation error between the frames \mathcal{F}_m and \mathcal{F}_r , we obtain

$$\begin{cases} \dot{x}_e = u_{2,r} y_e + u_1 \cos \theta_e - u_{1,r}, \\ \dot{y}_e = -u_{2,r} x_e + u_1 \sin \theta_e, \\ \dot{\theta}_e = u_2 - u_{2,r}. \end{cases} \quad (34.21)$$

To determine a control (u_1, u_2) which asymptotically stabilizes the error (x_e, y_e, θ_e) at zero, let us consider the following change of coordinates and control variables

$$(x_e, y_e, \theta_e, u_1, u_2) \mapsto (z_1, z_2, z_3, w_1, w_2)$$

defined by

$$\begin{aligned} z_1 &= x_e, \\ z_2 &= y_e, \\ z_3 &= \tan \theta_e, \\ w_1 &= u_1 \cos \theta_e - u_{1,r}, \\ w_2 &= \frac{u_2 - u_{2,r}}{\cos^2 \theta_e}. \end{aligned}$$

Note that, around zero, this mapping is only defined when $\theta_e \in (-\pi/2, \pi/2)$. In other words, the orientation error between the physical robot and the reference robot has to be smaller than $\pi/2$.

It is immediate to verify that, in the new variables, the system (34.21) can be written as

$$\begin{cases} \dot{z}_1 = u_{2,r} z_2 + w_1, \\ \dot{z}_2 = -u_{2,r} z_1 + u_{1,r} z_3 + w_1 z_3, \\ \dot{z}_3 = w_2. \end{cases} \quad (34.22)$$

We remark that the last term in each of the above three equations corresponds to the one of a chained system. We then have the following result:

Proposition 34.4

The control law

$$\begin{cases} w_1 = -k_1 |u_{1,r}| (z_1 + z_2 z_3) \quad (k_1 > 0), \\ w_2 = -k_2 u_{1,r} z_2 - k_3 |u_{1,r}| z_3, \quad (k_2, k_3 > 0), \end{cases} \quad (34.23)$$

renders the origin of the system (34.22) globally asymptotically stable if $u_{1,r}$ is a bounded differentiable function whose derivative is bounded and which does not tend to zero as t tends to infinity.

Remark

By comparison with the results of Sect. 34.3, we note that $u_{1,r}$ may well pass through zero and change its sign.

Proof

Consider the following positive-definite function

$$V(z) = \frac{1}{2} \left(z_1^2 + z_2^2 + \frac{1}{k_2} z_3^2 \right).$$

The time derivative of V along the trajectories of the controlled system (34.22–34.23) is given by

$$\dot{V} = z_1 w_1 + z_2 (u_{1,r} z_3 + w_1 z_3) + \frac{1}{k_2} z_3 w_2$$

$$\begin{aligned} &= w_1 (z_1 + z_2 z_3) + z_3 \left(u_{1,r} z_2 + \frac{1}{k_2} w_2 \right) \\ &= -k_1 |u_{1,r}| (z_1 + z_2 z_3)^2 - \frac{k_3}{k_2} |u_{1,r}| z_3^2. \end{aligned}$$

Therefore, along any of these trajectories, V is non-increasing and converges to some limit value $V_{\lim} \geq 0$. This implies that the variables z_1 , z_2 , and z_3 are bounded. Since $u_{1,r}$ is continuous, and since its derivative is bounded, $|u_{1,r}|$ is uniformly continuous. Therefore, \dot{V} is uniformly continuous and, by application of Barbalat's lemma, \dot{V} tends to zero when t tends to infinity. In view of the expression of \dot{V} , this implies that $u_{1,r} z_3$ and $u_{1,r} (z_1 + z_2 z_3)$ (and thus $u_{1,r} z_1$ also) tend to zero. On the other hand, by using the expression of $w_2 (= \dot{z}_3)$ one has

$$\frac{d}{dt} (u_{1,r}^2 z_3) = 2 \dot{u}_{1,r} u_{1,r} z_3 - k_3 u_{1,r}^2 |u_{1,r}| z_3 - k_2 u_{1,r}^3 z_2$$

and one deduces from what precedes that

$$\frac{d}{dt} (u_{1,r}^2 z_3) + k_2 u_{1,r}^3 z_2$$

tends to zero. Since $u_{1,r}^3 z_2$ is uniformly continuous (it is continuous and its derivative is bounded), and since $u_{1,r}^2 z_3$ tends to zero, one deduces by application of a slightly extended version of Barbalat's lemma that $u_{1,r}^3 z_2$ (and thus $u_{1,r} z_2$ also) tends to zero. In view of the expression of V , the convergence of $u_{1,r} z_i$ ($i = 1, 2, 3$) to zero implies the convergence of $u_{1,r} V$ to zero. Therefore, $u_{1,r} V_{\lim} = 0$, and $V_{\lim} = 0$ by using the assumption that $u_{1,r}$ does not tend to zero.

The linearization of the control (34.23) yields the simpler control

$$\begin{cases} w_1 = -k_1 |u_{1,r}| z_1, \\ w_2 = -k_2 u_{1,r} z_2 - k_3 |u_{1,r}| z_3. \end{cases}$$

Since $w_1 \approx u_1 - u_{1,r}$ and $w_2 \approx u_2 - u_{2,r}$ near the origin, it is legitimate to wonder if the control

$$\begin{cases} u_1 = u_{1,r} - k_1 |u_{1,r}| z_1, \\ u_2 = u_{2,r} - k_2 u_{1,r} z_2 - k_3 |u_{1,r}| z_3. \end{cases}$$

can also be used for the system (34.21). In fact, it is not difficult to verify, via a classical pole-placement calculation, that this control asymptotically stabilizes the origin of the linear system which approximates the system (34.21) when $u_{1,r}$ and $u_{2,r}$ are constant, with $u_{1,r} \neq 0$. Therefore, it also *locally* asymptotically stabilizes the origin of the system (34.21) when these conditions on

$u_{1,r}$ and $u_{2,r}$ are satisfied. This means also that the tuning of the gains $k_{1,2,3}$ can be performed by using classical linear control techniques applied to the linear approximation of the system (34.21). The control (34.23) is in fact designed so that its tuning for the specific velocities $u_{1,r} = 1$ and $u_{2,r} = 0$ gives good results for all other velocities (except, of course, $u_{1,r} = 0$, for which the linear approximation of the system (34.21) is not controllable and the control vanishes). Indeed, the multiplication of all control gains by $\pm u_{1,r}$ reduces to normalizing the equations of the controlled system with respect to the longitudinal velocity, so that the transient path followed by the vehicle in the process of catching up with the reference vehicle is independent of the intensity of the longitudinal velocity.

Generalization to a Car-Like Vehicle

The previous method extends to the car case. We provide below the main steps of this extension, and leave to the interested reader the task of verifying the details.

Consider the car's kinematic model (34.5), complemented with the following model of the reference car that one wishes to track

$$\begin{cases} \dot{x}_r = u_{1,r} \cos \theta_r, \\ \dot{y}_r = u_{1,r} \sin \theta_r, \\ \dot{\theta}_r = \frac{u_{1,r}}{L} \tan \phi_r, \\ \dot{\phi}_r = u_{2,r}. \end{cases} \quad (34.24)$$

We assume that there exists $\delta \in (0, \pi/2)$ such that the steering angle ϕ_r belongs to the interval $[-\delta, \delta]$.

By defining x_e, y_e , and θ_e as in the unicycle case, and by setting $\phi_e = \phi - \phi_r$, one easily obtains the following system (to be compared with (34.21))

$$\begin{cases} \dot{x}_e = (\frac{u_{1,r}}{L} \tan \phi_r) y_e + u_1 \cos \theta_e - u_{1,r}, \\ \dot{y}_e = -(\frac{u_{1,r}}{L} \tan \phi_r) x_e + u_1 \sin \theta_e, \\ \dot{\theta}_e = \frac{u_1}{L} \tan \phi - \frac{u_{1,r}}{L} \tan \phi_r, \\ \dot{\phi}_e = u_2 - u_{2,r}. \end{cases} \quad (34.25)$$

Introduce the new state variables

$$\begin{cases} z_1 = x_e, \\ z_2 = y_e, \\ z_3 = \tan \theta_e, \\ z_4 = \frac{\tan \phi - \cos \theta_e \tan \phi_r}{L \cos^3 \theta_e} + k_2 y_e, (k_2 > 0). \end{cases}$$

We note that for any $\phi_r \in (-\pi/2, \pi/2)$, the mapping $(x_e, y_e, \theta_e, \phi) \mapsto z$ defines a diffeomorphism between

$\mathbb{R}^2 \times (-\pi/2, \pi/2)^2$ and \mathbb{R}^4 . Introduce now the new control variables

$$\begin{cases} w_1 = u_1 \cos \theta_e - u_{1,r}, \\ w_2 = \dot{z}_4 = k_2 \dot{y}_e + \left(\frac{3 \tan \phi}{\cos \theta_e} - 2 \tan \phi_r \right) \frac{\sin \theta_e}{L \cos^3 \theta_e} \dot{\theta}_e \\ \quad - \frac{u_{2,r}}{L \cos^2 \phi_r \cos^2 \theta_e} + \frac{u_2}{L \cos^2 \phi \cos^3 \theta_e}. \end{cases} \quad (34.26)$$

One shows that $(u_1, u_2) \mapsto (w_1, w_2)$ defines a change of variables for θ_e, ϕ , and ϕ_r , inside the interval $(-\pi/2, \pi/2)$. These changes of state and control variables transform the system (34.25) into

$$\begin{cases} \dot{z}_1 = (\frac{u_{1,r}}{L} \tan \phi_r) z_2 + w_1, \\ \dot{z}_2 = -(\frac{u_{1,r}}{L} \tan \phi_r) z_1 + u_{1,r} z_3 + w_1 z_3, \\ \dot{z}_3 = -k_2 u_{1,r} z_2 + u_{1,r} z_4 \\ \quad + w_1 \left(z_4 - k_2 z_2 + (1 + z_3^2) \frac{\tan \phi_r}{L} \right), \\ \dot{z}_4 = w_2. \end{cases} \quad (34.27)$$

Proposition 34.4 then becomes:

Proposition 34.5

The control law

$$\begin{cases} w_1 = -k_1 |u_{1,r}| \left(z_1 + \frac{z_3}{k_2} \left[z_4 + (1 + z_3^2) \frac{\tan \phi_r}{L} \right] \right), \\ w_2 = -k_3 u_{1,r} z_3 - k_4 |u_{1,r}| z_4, \end{cases} \quad (34.28)$$

with $k_{1,2,3,4}$ denoting positive numbers, renders the origin of the system (34.27) globally asymptotically stable if (i) $u_{1,r}$ is a bounded differentiable function whose derivative is bounded and which does not tend to zero when t tends to infinity, and (ii) $|\phi_r|$ is smaller or equal to $\delta < \pi/2$.

As in the case of the unicycle, the gain parameters k_i can be tuned from the controlled system's linearization. More precisely, one can verify from (34.25), (34.26), and (34.28), that in the coordinates $\eta = (x_e, y_e, \theta_e, \phi_e/L)^\top$, the linearization of the controlled system at the equilibrium $\eta = 0$ yields, when $u_r = (1, 0)$, the linear system $\dot{\eta} = A\eta$ with

$$A = \begin{pmatrix} -k_1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -k_2 k_4 & -k_3 & -k_4 \end{pmatrix}$$

The control gains k_i can then be chosen to give desired values to the roots of the corresponding characteristic polynomial $P(\lambda) = (\lambda + k_1)(\lambda^3 + k_4\lambda^2 + k_3\lambda + k_2k_4)$. The nonlinear feedback law (34.28) is designed so that this choice also yields good results when the intensity of $u_{1,r}$ is different from 1 and/or varies arbitrarily, provided that the convergence conditions specified in Proposition 34.5 are satisfied.

The simulation shown on Fig. 34.7 illustrates this control scheme. The gain parameters k_i have been chosen as $(k_1, k_2, k_3, k_4) = (1, 1, 3, 3)$. The initial configuration of the reference vehicle (i.e. at $t = 0$), which is represented in Fig. 34.7a by dashed lines, is $(x_r, y_r, \theta_r)(0) = (0, 0, 0)$. The reference control u_r is defined by (34.29). The initial configuration of the controlled robot, represented in the figure by plain lines, is $(x, y, \theta)(0) = (0, -1.5, 0)$. The configurations at time $t = 10, 20$, and 30 , are also represented on the figure. Due to the fast convergence of the tracking error to zero (see the time evolution of the components x_e, y_e, θ_e of the tracking error in Fig. 34.7b), one can basically consider that the configurations of both vehicles coincide after time $t = 10$.

$$u_r(t) = \begin{cases} (1, 0)^\top, & \text{if } t \in [0, 10], \\ (-1, 0.5 \cos(2\pi(t-10)/5))^\top, & \text{if } t \in [10, 20], \\ (1, 0)^\top, & \text{if } t \in [20, 30]. \end{cases} \quad (34.29)$$

34.4.3 Path Following with Orientation Control

We reconsider the path-following problem with the reference point P now located on the actuated wheels axle, at the mid-distance of the wheels. The objective is to synthesize a control law which allows the vehicle to follow the path in a stable manner, independently of the sign of the longitudinal velocity.

Unicycle Case

We have seen in Sect. 34.4.1 how to transform kinematic equations with respect to a Frénet frame into the three-dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1, \\ \dot{z}_2 = v_1 z_3, \\ \dot{z}_3 = v_2. \end{cases} \quad (34.30)$$

Recall that $(z_1, z_2, z_3) = (s, d, (1 - dc(s)) \tan \theta_e)$ and that $v_1 = u_1 / (1 - dc(s) \cos \theta_e)$. The objective is to determine a control law which asymptotically stabilizes $(d = 0, \theta_e = 0)$ and also ensures that the constraint on the distance d to the path (i.e., $|dc(s)| < 1$) is satisfied along the trajectories of the controlled system. For the control law, a first possibility consists in considering a proportional feedback like

$$v_2 = -v_1 k_2 z_2 - |v_1| k_3 z_3, \quad (k_2, k_3 > 0). \quad (34.31)$$

It is then immediate to verify that the origin of the closed-loop subsystem

$$\begin{cases} \dot{z}_2 = v_1 z_3, \\ \dot{z}_3 = -v_1 k_2 z_2 - |v_1| k_3 z_3 \end{cases} \quad (34.32)$$

is asymptotically stable when v_1 is constant, either positive or negative. Since u_1 (not v_1) is the intensity of

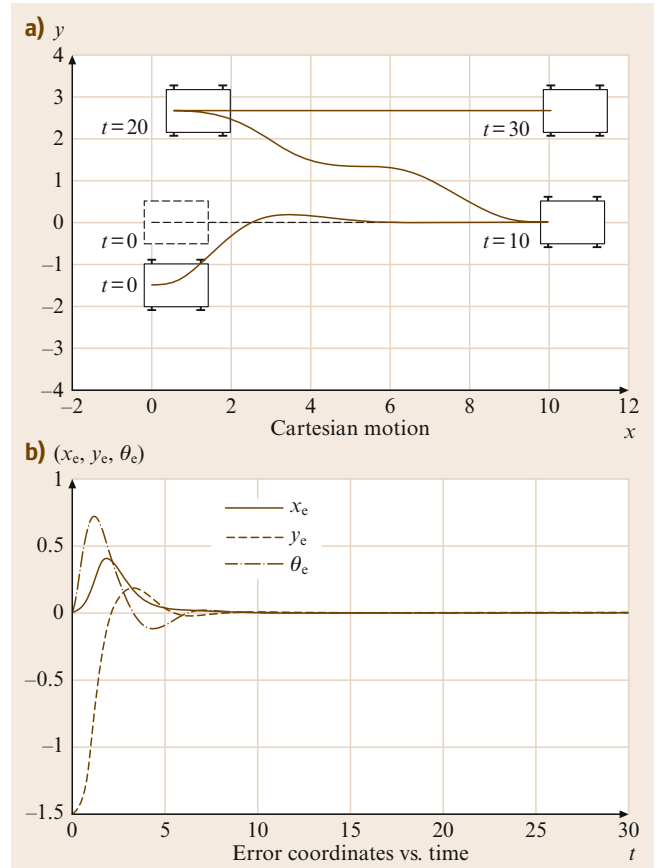


Fig. 34.7a,b Tracking of a reference vehicle

the vehicle's longitudinal velocity, one would rather establish stability conditions which depend on u_1 . The following result provides a rather general stability condition and gives a sufficient condition for the satisfaction of the constraint $|dc(s)| < 1$.

Proposition 34.6

Consider the system (34.30) controlled with (34.31), and assume that the initial conditions $[z_2(0), z_3(0)] = [d(0), \{1 - d(0)c[s(0)]\} \tan \theta_e(0)]$ verify

$$z_2^2(0) + \frac{1}{k_2} z_3^2(0) < \frac{1}{c_{\max}^2}$$

with $c_{\max} = \max_s |c(s)|$. Then, the constraint $|dc(s)| < 1$ is satisfied along any solution to the controlled system. Moreover, the function

$$V(z) = \frac{1}{2} \left(z_2^2 + \frac{1}{k_2} z_3^2 \right) \quad (34.33)$$

is nonincreasing along any trajectory $z(t)$ of the system, and $V[z(t)]$ tends to zero as t tends to infinity if, for instance, u_1 is a bounded differentiable time function whose derivative is bounded and which does not tend to zero as t tends to infinity.

The proof is similar to the one of Proposition 34.4: a simple calculation shows that the function V is nonincreasing so that, along any solution to the controlled system, it converges to some limit value V_{\lim} . The same arguments as those used in the proof of Proposition 34.4 can then be repeated to show that $V_{\lim} = 0$.

Note that the constraints upon u_1 are rather weak. In particular, the sign of u_1 does not have to be constant.

From a practical point of view it can be useful to complement the control action with an integral term. More precisely, let us define a variable z_0 by

$$\dot{z}_0 = v_1 z_2, \quad z_0(0) = 0.$$

The control (34.31) can be modified as follows:

$$\begin{aligned} v_2 &= -|v_1|k_0 z_0 - v_1 k_2 z_2 - |v_1|k_3 z_3, \\ &= -|v_1|k_0 \int_0^t v_1 z_2 - v_1 k_2 z_2 - |v_1|k_3 z_3, \\ &\quad (k_0, k_2, k_3 > 0) \end{aligned} \quad (34.34)$$

and Proposition 34.6 becomes:

Proposition 34.7

Consider the system (34.30) controlled by (34.34) with k_0, k_2 , and k_3 such that the polynomial

$$s^3 + k_3 s^2 + k_2 s + k_0$$

is Hurwitz stable (all roots of this polynomial have a negative real part.). Assume also that the initial conditions $[z_2(0), z_3(0)] = [d(0), \{1 - d(0)c[s(0)]\} \tan \theta_e(0)]$ verify:

$$z_2^2(0) + \frac{1}{k_2 - \frac{k_0}{k_3}} z_3^2(0) < \frac{1}{c_{\max}^2}.$$

Then the constraint $|dc(s)| < 1$ is satisfied along any solution to the controlled system. Moreover, the function

$$\frac{k_0}{k_3} \left(\int_0^t v_1 z_2 \right)^2 + z_2^2(t) + \frac{1}{k_2 - \frac{k_0}{k_3}} z_3^2(t)$$

is nonincreasing along any trajectory of the system, and it tends to zero as t tends to infinity if, for instance, u_1 is a bounded differentiable time-function whose derivative is bounded and which does not converge to zero as t tends to infinity.

Generalization to a Car-Like Vehicle and to a Unicycle-Type Vehicle with Trailers

One of the assets of this type of approach, besides the simplicity of the control law and little demanding conditions of stability associated with it, is that it can be generalized in a straightforward manner to car-like vehicles and unicycle-type vehicles with trailers. The result is summarized in the next proposition by considering a n -dimensional chained system

$$\begin{cases} \dot{z}_1 = v_1, \\ \dot{z}_2 = v_1 z_3, \\ \vdots \\ \dot{z}_{n-1} = v_1 z_n, \\ \dot{z}_n = v_2, \end{cases} \quad (34.35)$$

with $n \geq 3$. Its proof is a direct extension of the one in the three-dimensional case. The dimension $n = 4$ corresponds to the car case (Sect. 34.4.1). As for a unicycle-type vehicle with N trailers, one has $n = N + 3$. Recall also that, in all cases, z_2 represents the distance d between the path and the point P located at the mid-distance of the rear wheels of the last vehicle.

Proposition 34.8

Let k_2, \dots, k_n denote parameters such that the polynomial

$$s^{n-1} + k_n s^{n-2} + k_{n-1} s^{n-3} + \dots + k_3 s + k_2$$

is Hurwitz stable. With these parameters, let us associate the control law

$$v_2 = -v_1 \sum_{i=2}^n \text{sign}(v_1)^{n+1-i} k_i z_i. \quad (34.36)$$

Then, there exists a positive-definite matrix \mathbf{Q} (whose entries depend on the coefficients k_i) such that, if the initial conditions $[z_2(0), z_3(0), \dots, z_n(0)]$ verify

$$\|[z_2(0), z_3(0), \dots, z_n(0)]\|_{\mathbf{Q}} < \frac{1}{c_{\max}} \quad (34.37)$$

the constraint $|dc(s)| < 1$ is satisfied along any solution to the controlled system. Moreover, the function

$$\|[z_2(t), z_3(t), \dots, z_n(t)]\|_{\mathbf{Q}}$$

is nonincreasing along any trajectory of the system, and it tends to zero as t tends to infinity if, for instance, u_1 is a bounded differentiable function whose derivative is bounded and which does not converge to zero as t tends to infinity.

Remark

The condition (34.37) is always satisfied when $c(s) = 0$ for every s (i.e., when the path is a straight line). It is little demanding in practice when c_{\max} is small. Note also that it is possible to calculate the matrix \mathbf{Q} explicitly as a function of the parameters k_i (see [34.1] for more details).

As in the three-dimensional case, it is possible to add an integral term to the control. In this case, the control is calculated from the expression of an *extended system* whose state vector is composed of the variables z_1, \dots, z_n , and a complementary variable z_0 such that $\dot{z}_0 = v_1 z_2$. Since adding the variable z_0 preserves the chained structure of the system, the control expression is simply adapted from the one determined for a system of dimension $n + 1$ with no integral term. More precisely, one obtains

$$v_2 = -k_0 v_1 \text{sign}(v_1)^n \times \int_0^t v_1 z_2 - v_1 \sum_{i=2}^n \text{sign}(v_1)^{n+1-i} k_i z_i$$

with the parameters k_i chosen so that the polynomial $s^n + k_n s^{n-1} + k_{n-1} s^{n-2} + \dots + k_2 s + k_0$ is Hurwitz stable.

The simulation results reported in Fig. 34.8 illustrate how this control scheme performs for a car-like vehicle. The reference curve is the circle of radius equal to four, centered at the origin. The robot's longitudinal velocity u_1 is defined by $u_1 = 1$ for $t \in [0, 5]$,

and $u_1 = -1$ for $t > 5$. The control gains have been chosen as $(k_2, k_3, k_4) = (1, 3, 3)$. The motion of the car-like robot in the plane is represented in Fig. 34.8a, and its configuration at times $t = 0, 5$, and 25 are also depicted in the figure. The time evolution of the variables z_2, z_3, z_4 (defined in Proposition 34.3) is represented in Fig. 34.8b. One can observe that the (discontinuous) change of the longitudinal velocity u_1 at $t = 5$ does not affect the convergence of these variables to zero.

34.4.4 Asymptotic Stabilization of Fixed Postures

We now consider the problem of asymptotic stabilization of a fixed desired (reference) posture (i.e., position

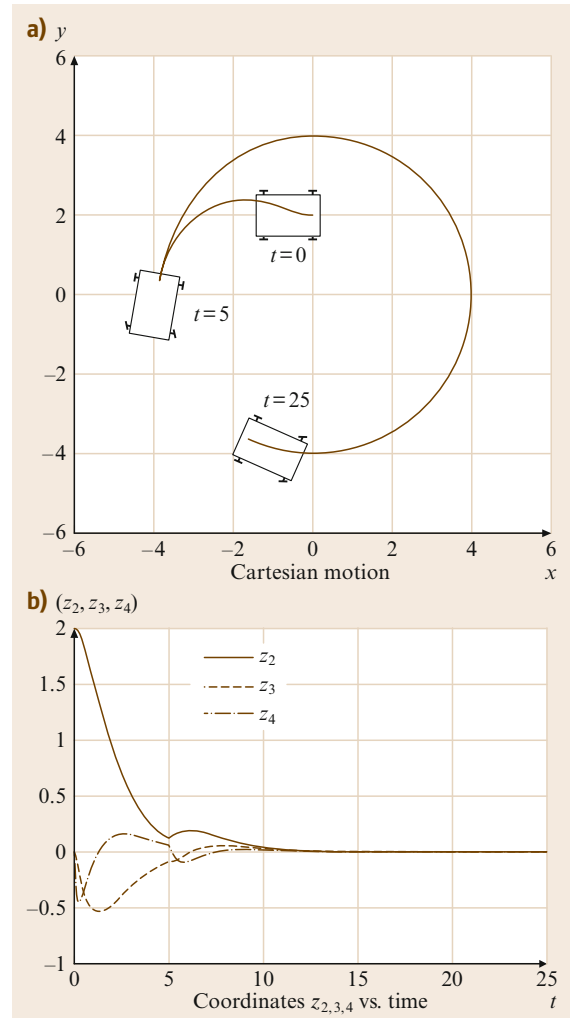


Fig. 34.8a,b Path following along a circle

and orientation) for the robot chassis. This problem may be seen as a limit case of the trajectory tracking problem. However, none of the feedback controllers proposed previously in this chapter provides a solution to this problem. For instance, in Sect. 34.3, although the result about the stabilization in position of feasible trajectories did not exclude the case of a trajectory reduced to a single point, the stability of the vehicle's orientation was not granted in this case. As for the results in Sect. 34.4, the convergence of the posture error to zero has been proven when the robot's longitudinal velocity did not converge to zero (which excludes the case of fixed postures).

From the automatic control point of view, the asymptotic stabilization of fixed postures is very different from the problems of path following and trajectory tracking with nonzero longitudinal velocity, much in the same way as a human driver knows, from experience, that parking a car at a precise location involves techniques and skills different from those exercised when cruising on a road. In particular, it cannot be solved by any classical control method for linear systems (or based on linearization). Technically, the underlying general problem is the one of asymptotic stabilization of equilibria of controllable driftless systems with less control inputs than state variables. This problem motivated numerous studies during the last decade of the last century, from many authors and with various angles of attack, and it has remained a subject of active research five years later. The variety of candidate solutions proposed until now, the mathematical technicalities associated with several of them, together with unsolved difficulties and limitations, particularly (but not only) in terms of robustness (an issue on which we will return), prevent us from attempting to cover the subject exhaustively. Instead, we have opted for a somewhat informal exposition of approaches which have been considered, with the illustration of a few control solutions, without going into technical and mathematical details.

A central aspect of the problem, which triggered much of the subsequent research on the control of nonholonomic systems, is that asymptotic stabilization of equilibria (or fixed points) cannot be achieved by using continuous feedbacks which depend on the state only (i. e., continuous pure-state feedbacks). This is a consequence of an important result due to Brockett in 1983 (see also related comments in Sect. 17.4.2). The original result by Brockett concerned differentiable feedbacks; it has later been extended to the larger set of feedbacks which are only continuous.

Theorem 34.1

(Brockett [34.2]) Consider a control system $\dot{x} = f(x, u)$ ($x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$), with f a differentiable function and $(x, u) = (0, 0)$ an equilibrium of this system. A necessary condition for the existence of a continuous feedback control $u(x)$ which renders the origin of the closed-loop system $\dot{x} = f(x, u(x))$ asymptotically stable is the local surjectivity of the application $(x, u) \mapsto f(x, u)$. More precisely, the image by f of any neighborhood Ω of $(0, 0)$ in \mathbb{R}^{n+m} must be a neighborhood of 0 in \mathbb{R}^n .

This result implies that the equilibria of many controllable (nonlinear) systems are not asymptotically stabilizable by continuous pure-state feedbacks. All nonholonomic WMRs belong to this category of systems. This will be shown in the case of a unicycle-type vehicle; the proof for the other mobile robots is similar. Let us thus consider a unicycle-type vehicle, whose kinematic equations (34.4) can be written as $\dot{x} = f(x, u)$ with $x = (x_1, x_2, x_3)^\top$, $u = (u_1, u_2)^\top$, and $f(x, u) = (u_1 \cos x_3, u_1 \sin x_3, u_2)^\top$, and let us show that f is not locally onto in the neighborhood of $(x, u) = (0, 0)$. To this purpose, take a vector in \mathbb{R}^3 of the form $(0, \delta, 0)^\top$. It is obvious that the equation $f(x, u) = (0, \delta, 0)^\top$ does not have a solution in the neighborhood of $(x, u) = (0, 0)$ since the first equation, namely $u_1 \cos x_3 = 0$, implies that $u_1 = 0$, so that the second equation cannot have a solution if δ is different from zero.

It is also obvious that the linear approximation (about the equilibrium $(x, u) = (0, 0)$) of the unicycle kinematic equations is not controllable. If it were, it would be possible to (locally) asymptotically stabilize this equilibrium with a linear (thus continuous) state feedback.

Therefore, by application of the above theorem, a unicycle-type mobile robot (like other nonholonomic robots) cannot be asymptotically stabilized at a desired posture (position/orientation) by using a continuous pure-state feedback. This impossibility has motivated the development of other control strategies in order to solve the problem. Three major types of controls have been considered:

1. *continuous time-varying feedbacks*, which, besides from depending on the state x , depend also on the exogenous time variable (i. e., $u(x, t)$ instead of $u(x)$ for classical feedbacks),
2. *discontinuous feedbacks*, in the classical form $u(x)$, except that the function u is not continuous at the equilibrium that one wishes to stabilize,

3. *hybrid discrete/continuous feedbacks*. Although this class of feedbacks is not defined as precisely as the other two sets of controls, it is mostly composed of time-varying feedbacks, either continuous or discontinuous, such that the part of the control which depends upon the state is only updated periodically, e.g., $\mathbf{u}(t) = \bar{\mathbf{u}}[\mathbf{x}(kT), t]$ for any $t \in [kT, (k+1)T]$, with T denoting a constant period, and $k \in \mathbb{N}$.

We will now illustrate these approaches. In fact, only time-varying and hybrid feedbacks will be considered here. The main reason is that discontinuous feedbacks involve difficult questions (existence of solutions, mathematical meaning of these solutions, etc.) which complicate their analysis and for which complete answers are not available. Moreover, for most of the discontinuous control strategies described in the literature, the property of stability in the sense of Lyapunov is either not granted or remains an open issue.

Time-Varying Feedbacks

The use of time-varying feedbacks for the asymptotic stabilization of a fixed desired equilibrium, for a nonholonomic WMR, in order to circumvent the obstruction pointed out by Brockett's Theorem, was first proposed in [34.3]. Since then, very general results about the stabilization of nonlinear systems by means of time-varying feedbacks have been obtained. For instance, it has been proved that any controllable driftless system can have any of its equilibria asymptotically stabilized with a control of this type [34.4]. This includes the kinematic models of the nonholonomic mobile robots here considered. We will illustrate this approach in the case of unicycle-type and car-like mobile robots modeled by three- and four-dimensional chained systems, respectively. In order to consider the three-dimensional case, let us come back on the results obtained in Sect. 34.4.3 for path following. We have established (Proposition 34.6) that the control $v_2 = -v_1 k_2 z_2 - |v_1| k_3 z_3$ applied to the system

$$\begin{cases} \dot{z}_1 = v_1, \\ \dot{z}_2 = v_1 z_3, \\ \dot{z}_3 = v_2, \end{cases}$$

renders the function $V(z)$ defined by (34.33) nonincreasing along any trajectory of the controlled system, i. e.,

$$\dot{V} = -\frac{k_3}{k_2} |v_1| z_3^2,$$

and ensures the convergence of z_2 and z_3 to zero if, for instance, v_1 does not tend to zero as t tends to infinity. For example, if $v_1(t) = \sin t$, the proposition applies, z_2 and z_3 tend to zero, and

$$\begin{aligned} z_1(t) &= z_1(0) + \int_0^t v_1(s) ds = z_1(0) + \int_0^t \sin s ds \\ &= z_1(0) + 1 - \cos t \end{aligned}$$

so that $z_1(t)$ oscillates around the mean value $z_1(0) + 1$. To reduce these oscillations, one can multiply v_1 by a factor which depends on the current state. Take, for example, $v_1(z, t) = \|(z_2, z_3)\| \sin t$, that we complement with a stabilizing term like $-k_1 z_1$ with $k_1 > 0$, i. e.,

$$v_1(z, t) = -k_1 z_1 + \|(z_2, z_3)\| \sin t.$$

The feedback control so obtained is time-varying and asymptotically stabilizing.

Proposition 34.9

The continuous time-varying feedback

$$\begin{cases} v_1(z, t) = -k_1 z_1 + \alpha \|(z_2, z_3)\| \sin t, \\ v_2(z, t) = -v_1(z, t) k_2 z_2 - |v_1(z, t)| k_3 z_3, \end{cases} \quad (34.38)$$

with $\alpha, k_{1,2,3} > 0$, renders the origin of the three-dimensional chained system globally asymptotically stable [34.5].

The above proposition can be extended to chained systems of arbitrary dimension [34.5]. For the case $n = 4$, which corresponds to the car-like robot, one has the following result.

Proposition 34.10

The continuous time-varying feedback

$$\begin{cases} v_1(z, t) = -k_1 z_1 + \alpha \|(z_2, z_3, z_4)\| \sin t, \\ v_2(z, t) = -|v_1(z, t)| k_2 z_2 - v_1(z, t) k_3 z_3 \\ \quad - |v_1(z, t)| k_4 z_4, \end{cases} \quad (34.39)$$

with $\alpha, k_{1,2,3,4} > 0$ chosen such that the polynomial $s^3 + k_4 s^2 + k_3 s + k_2$ is Hurwitz-stable, renders the origin of the four-dimensional chained system globally asymptotically stable [34.5].

Figure 34.9 below illustrates the previous result. For this simulation, the parameters $\alpha, k_{1,2,3,4}$ in the feedback law (34.39) have been chosen as $\alpha = 3$ and $k_{1,2,3,4} = (1.2, 10, 18, 17)$. Figure 34.9a shows the motion of the

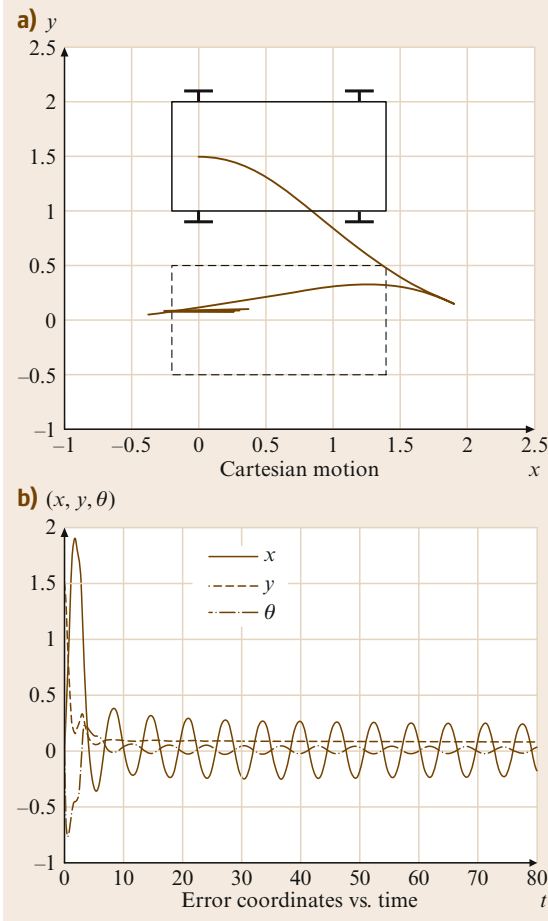


Fig. 34.9a,b Asymptotic stabilization with a Lipschitz-continuous controller

car-like robot in the plane. The initial configuration, at time $t = 0$, is depicted in plain lines, whereas the desired configuration is shown in dashed lines. The time evolution of the variables x , y , and θ (i. e., the position and orientation variables corresponding to the model (34.5)) is shown in Fig. 34.9b.

A shortcoming of this type of control, very clear from this simulation, is that the system's state converges to zero quite slowly. One can show that the rate of convergence is only polynomial, i. e., it is commensurable with $t^{-\alpha}$ (for some $\alpha \in (0, 1)$) for most of the trajectories of the controlled system. This slow rate of convergence is related to the fact that the control function is Lipschitz-continuous with respect to x . It is a characteristic of systems the linear approximation of which is not stabilizable, as specified in the following proposition.

Proposition 34.11

Consider the control system $\dot{x} = f(x, u)$ ($x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$) with f being differentiable, and $(x, u) = (0, 0)$ an equilibrium point of this system. Assume that the linear approximation of this system is not stabilizable. Consider also a continuous time-varying feedback $u(x, t)$, periodic with respect to t , such that $u(0, t) = 0$ for any t , and such that $u(\cdot, t)$ is $k(t)$ -Lipschitz continuous with respect to x , for some bounded function k . This feedback cannot yield uniform exponential convergence to zero of the closed-loop system's solutions: there do not exist constants $K > 0$ and $\gamma > 0$ such that, along any trajectory $x(\cdot)$ of the controlled system, one has

$$|x(t)| \leq K|x(t_0)|e^{-\gamma(t-t_0)}. \quad (34.40)$$

The intuitive reason behind this impossibility can easily be illustrated on the unicycle example. When using the chain form representation, the second equation is $\dot{z}_2 = v_1 z_3$. Since the linearization around $(z = 0, v = 0)$ of this equation gives $\dot{z}_2 = 0$, the linear approximation of the system is not controllable (nor stabilizable). In these conditions, exponential convergence, when applying a linear feedback, would necessitate the use of gains growing to infinity, thus ruling out the property of Lipschitz-continuity. This type of reasoning, coupled to the need of better performance and efficiency, has triggered the development of stabilizing time-varying feedbacks which are continuous, but not Lipschitz-continuous. Examples of such feedbacks, yielding uniform exponential convergence, are given in the following propositions for chained systems of dimension three and four, respectively.

Proposition 34.12

Let $\alpha, k_{1,2,3} > 0$ denote scalars such that the polynomial $p(s) = s^2 + k_3 s + k_2$ is Hurwitz stable. For any integers $p, q \in \mathbb{N}^*$, let $\rho_{p,q}$ denote the function defined on \mathbb{R}^2 by

$$\begin{aligned} \forall \bar{z}_2 = (z_2, z_3) \in \mathbb{R}^2, \rho_{p,q}(\bar{z}_2) \\ = \left(|z_2|^{\frac{p}{q+1}} + |z_3|^{\frac{p}{q}} \right)^{\frac{1}{p}}. \end{aligned}$$

Then, there exists $q_0 > 1$ such that, for any $q \geq q_0$ and $p > q + 2$, the continuous state feedback

$$\begin{cases} v_1(z, t) = -k_1(z_1 \sin t - |z_1|) \sin t \\ \quad + \alpha \rho_{p,q}(\bar{z}_2) \sin t \\ v_2(z, t) = -v_1(z, t) k_2 \frac{z_2}{\rho_{p,q}^2(\bar{z}_2)} - |v_1(z, t)| k_3 \frac{z_3}{\rho_{p,q}(\bar{z}_2)} \end{cases} \quad (34.41)$$

renders the origin of the three-dimensional chained system globally asymptotically stable, with a uniform exponential rate of convergence [34.1].

The parenthood of the controls (34.38) and (34.41) is noticeable. One can also verify that the control (34.41) is well defined (by continuity) at $\bar{z}_2 = 0$. More precisely, the ratios

$$\frac{z_2}{\rho_{p,q}^2(\bar{z}_2)} \quad \text{and} \quad \frac{z_3}{\rho_{p,q}(\bar{z}_2)},$$

which are obviously well defined when $\bar{z}_2 \neq 0$, tend to zero when \bar{z}_2 tends to zero. This guarantees the continuity of the control law.

The property of exponential convergence pointed out in the above result calls for some remarks. Indeed, this property does not exactly correspond to the classical exponential convergence property associated with stable linear systems. In this latter case, exponential convergence implies that the relation (34.40) is satisfied. This corresponds to the common notion of *exponential stability*. In the present case, this inequality becomes

$$\rho[z(t)] \leq K \rho[z(t_0)] e^{-\gamma(t-t_0)}$$

for some function ρ , defined for example by $\rho(z) = |z_1| + \rho_{p,q}(z_2, z_3)$, with $\rho_{p,q}$ as specified in Proposition 34.12. Although the function ρ shares common features with the Euclidean norm of the state vector (it is definite positive and it tends to infinity when $\|z\|$ tends to infinity), it is not equivalent to this norm. Of course, this does not change the fact that each component z_i of z converges to zero exponentially. However, the transient behavior is different because one only has

$$|z_i(t)| \leq K \|z(t_0)\|^\alpha e^{-\gamma(t-t_0)}$$

with $\alpha < 1$, instead of

$$|z_i(t)| \leq K \|z(t_0)\| e^{-\gamma(t-t_0)}.$$

In the case of the four-dimensional chained system, one can establish the following result, which is similar to Proposition 34.12.

Proposition 34.13

Let $\alpha, k_1, k_2, k_3, k_4 > 0$ be chosen such that the polynomial $p(s) = s^3 + k_4 s^2 + k_3 s + k_2$ is Hurwitz stable. For any integers $p, q \in \mathbb{N}^*$, let $\rho_{p,q}$ denote the function defined on \mathbb{R}^3 by

$$\rho_{p,q}(\bar{z}_2) = \left(|z_2|^{\frac{p}{q+2}} + |z_3|^{\frac{p}{q+1}} + |z_4|^{\frac{p}{q}} \right)^{\frac{1}{p}}$$

with $\bar{z}_2 = (z_2, z_3, z_4) \in \mathbb{R}^3$. Then, there exists $q_0 > 1$ such that, for any $q \geq q_0$ and $p > q + 2$, the continuous state feedback

$$\begin{cases} v_1(z, t) = -k_1(z_1 \sin t - |z_1|) \sin t \\ \quad + \alpha \rho_{p,q}(\bar{z}_2) \sin t, \\ v_2(z, t) = -|v_1(z, t)| k_2 \frac{z_2}{\rho_{p,q}^3(\bar{z}_2)} - v_1(z, t) k_3 \frac{z_3}{\rho_{p,q}^2(\bar{z}_2)} \\ \quad - |v_1(z, t)| k_4 \frac{z_4}{\rho_{p,q}(\bar{z}_2)}, \end{cases} \quad (34.42)$$

renders the origin of the four-dimensional chained system globally asymptotically stable, with a uniform exponential rate of convergence [34.1].

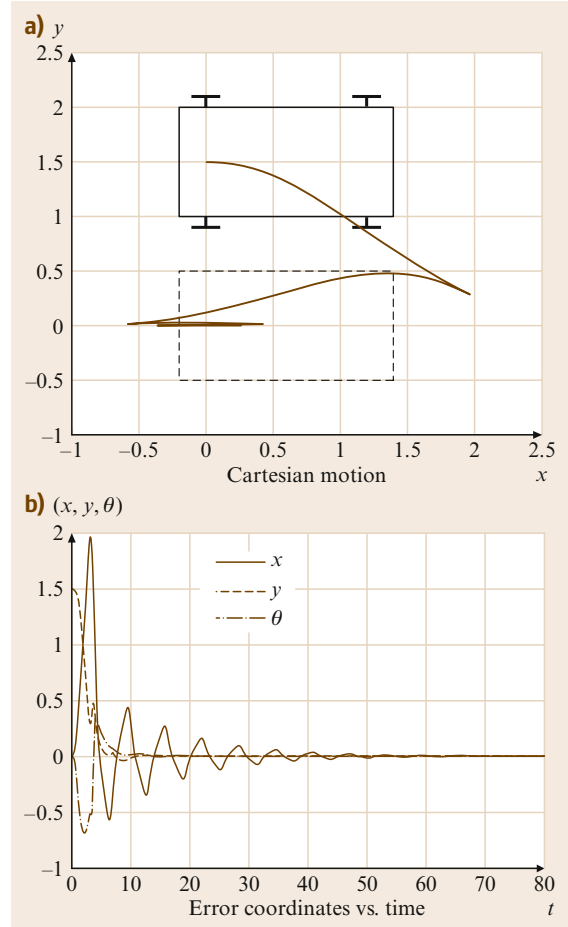


Fig. 34.10a,b Asymptotic stabilization with a continuous (non-Lipschitz) time-varying feedback

The performance of the control law (34.42) is illustrated by the simulation results shown in Fig. 34.10. The control parameters have been chosen as follows: $\alpha = 0.6$, $k_{1,2,3,4} = (1.6, 10, 18, 17)$, $q = 2$, $p = 5$. The comparison with the simulation results of Fig. 34.9 shows a clear gain in performance.

Hybrid Feedbacks

These feedbacks constitute an alternative for the asymptotic stabilization of fixed postures. They may be seen as a mix of open-loop and feedback controls in the sense that the dependence on the state is, in general, only updated periodically (by contrast with time-varying feedbacks which are updated continuously). Between two updates, the control works in open-loop mode. Nonetheless, this type of control may present some advantages with respect to time-varying feedbacks. This point is briefly commented upon a little further. An example of a hybrid feedback is provided in the following proposition.

Proposition 34.14

The hybrid feedback law v defined by

$$v(t) = \bar{v}[z(kT), t], \quad \forall t \in [kT, (k+1)T], \quad (34.43)$$

with

$$\begin{cases} \bar{v}_1(z, t) = \frac{1}{T}[(k_1 - 1)z_1 + 2\pi\rho(z)\sin(\omega t)], \\ \bar{v}_2(z, t) = \frac{1}{T}[(k_3 - 1)z_3 + 2(k_2 - 1)\frac{z_2}{\rho(z)}\cos(\omega t)], \end{cases}$$

and

$$k_{1,2,3} \in (-1, 1), \quad \omega = \frac{2\pi}{T}, \quad \rho(z) = \alpha_2|z_2|^{1/2}, \quad (\alpha_2 > 0)$$

is a $\mathcal{K}(T)$ -exponential stabilizer for the three-dimensional chained system [34.6].

The property of $\mathcal{K}(T)$ -exponential stabilizer evoked in the above proposition means that there exist positive constants K , η , and γ , with $\gamma < 1$, such that for any z_0 , the solution at time t of the controlled system associated with the initial condition z_0 at time $t = 0$, which we denote as $z(t, 0, z_0)$, satisfies for any $k \in \mathbb{N}$ and any $s \in [0, T)$ the inequalities

$$\|z((k+1)T, 0, z_0)\| \leq \gamma \|z(kT, 0, z_0)\|$$

and

$$\|z(kT + s, 0, z_0)\| \leq K \|z(kT, 0, z_0)\|^\eta.$$

These relations imply the exponential convergence of the system's trajectories to the origin $z = 0$. They do not imply the stability of this point because $\|z(t, 0, z_0)\|$ may vanish at some time $t = \bar{t}$ and not remain equal to zero thereafter. Note, however, that if $\|z(kT, 0, z_0)\| = 0$ for some $k \in \mathbb{N}$, then the above relations imply that $\|z(t, 0, z_0)\| = 0$ for all $t \geq kT$.

In the case of four-dimensional systems, a result similar to Proposition 34.14 can also be established.

Proposition 34.15

The hybrid feedback law v defined by

$$v(t) = \bar{v}(z(kT), t), \quad \forall t \in [kT, (k+1)T] \quad (34.44)$$

with

$$\begin{cases} \bar{v}_1(z, t) = \frac{1}{T}[(k_1 - 1)z_1 + 2\pi\rho(z)\sin(\omega t)], \\ \bar{v}_2(z, t) = \frac{1}{T}[(k_4 - 1)z_4 + 2(k_3 - 1)\frac{z_3}{\rho(z)}\cos(\omega t) \\ \quad + 8(k_2 - 1)\frac{z_2}{\rho^2(z)}\cos(2\omega t)], \end{cases}$$

$k_{1,2,3,4} \in (-1, 1)$, $\omega = \frac{2\pi}{T}$, and $\rho(z) = \alpha_2|z_2|^{1/4} + \alpha_3|z_3|^{1/3}$ ($\alpha_{2,3} > 0$), is a $\mathcal{K}(T)$ -exponential stabilizer for the four-dimensional chained system [34.6].

The simulation results reported in Fig. 34.11 illustrate the application of the feedback law (34.44). The control parameters have been chosen as $T = 3$, $k_{1,2,3,4} = 0.25$, and $\alpha_{2,3} = 0.95$. The control performance is similar to the one observed in Fig. 34.10, as could be expected from the fact that both controls yield exponential convergence to the origin.

34.4.5 Limitations Inherent to the Control of Nonholonomic Systems

Let us first mention some problems associated with the nonlinear time-varying and hybrid feedbacks just presented. An ever important issue, when studying feedback control, is robustness. Indeed, if it were not for the sake of robustness, feedback control would lose much of its value and interest with respect to open-loop control solutions. There are various robustness issues. One of them concerns the sensitivity to modeling errors. For instance, in the case of a unicycle-type robot whose kinematic equations are in the form $\dot{x} = u_1 b_1(x) + u_2 b_2(x)$, one would like to know whether a feedback law which stabilizes an equilibrium of this system also stabilizes this equilibrium for the *neighbor* system $\dot{x} = u_1[b_1(x) + \varepsilon g_1(x)] + u_2[b_2(x) + \varepsilon g_2(x)]$, with g_1 and g_2 denoting continuous applications, and ε a parameter which quantifies the modeling error. This type of error can account,

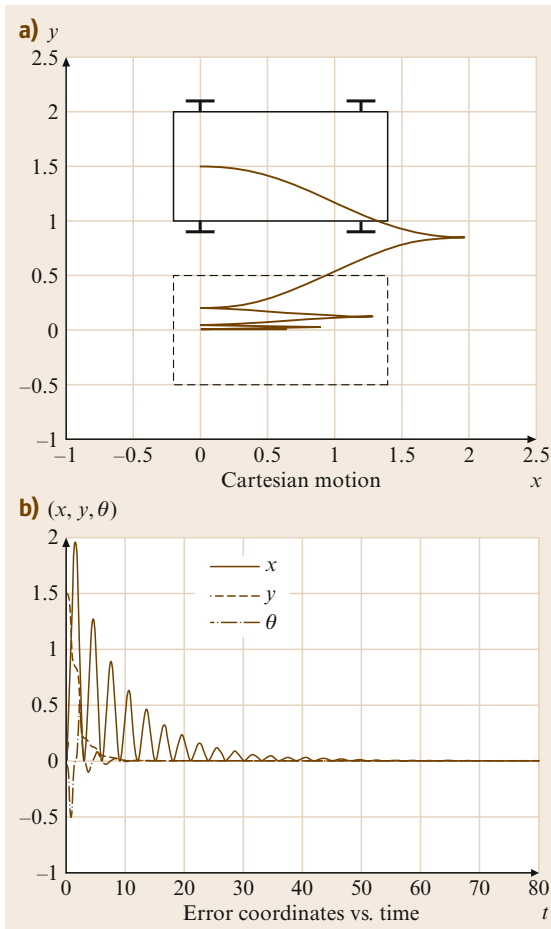


Fig. 34.11a,b Asymptotic stabilization with a hybrid discrete/continuous controller

for example, for a small uncertainty concerning the orientation of the actuated wheels axle with respect to the chassis, which results in a bias in the measurement of this orientation. One can show that time-varying control laws like (34.41) are not robust with respect to this type of error in the sense that, for certain functions g_1 and g_2 , and for ε arbitrarily small, the system's solutions end up oscillating in the neighborhood of the origin, instead of converging to the origin. In other words, both the properties of stability of the origin and of convergence to this point can be jeopardized by arbitrarily small modeling errors, even in the absence of measurement noise. In this respect, the hybrid control law (34.43) is more robust: the exponential convergence to the origin of the controlled system's solutions is still obtained when ε is small enough. However, the slightest discretization un-

certainly can produce the same type of local instability. In view of these problems, one is brought to question the existence of fast (exponential) stabilizers endowed with robustness properties similar to those of stabilizing linear feedbacks for linear systems. The answer is that, to our knowledge, no such control solution (either continuous or discontinuous) has ever been found. More than likely such a solution does not exist for nonholonomic systems. Robustness of the stability property against modeling errors, and control discretization and delays, has been proved in some cases, but this could only be achieved with Lipschitz-continuous feedbacks which, as we have seen, yield slow convergence. The classical compromise between robustness and performance thus seems much more acute than in the case of stabilizable linear systems (or nonlinear systems whose linear approximation is stabilizable).

A second issue is the proven nonexistence of a *universal* feedback controller capable of stabilizing any feasible reference state trajectory asymptotically [34.7]. This is another notable difference with the linear case. Indeed, given a controllable linear system $\dot{x} = Ax + Bu$, the feedback controller $u = u_r + K(x - x_r)$, with K a gain matrix such that $A + BK$ is Hurwitz stable, exponentially stabilizes any feasible reference trajectory x_r (solution to the system) associated with the control input u_r . The nonexistence of such a controller, in the case of nonholonomic mobile robots, is related to the conditions upon the longitudinal velocity stated in previous propositions concerning trajectory stabilization (Propositions 34.4 and 34.5). This basically indicates that such conditions cannot be removed entirely: whatever the chosen feedback controller, there always exists a feasible reference trajectory that this feedback cannot asymptotically stabilize. Note that this limitation persists when considering nonstandard feedbacks (such as, e.g., time-varying periodic feedbacks capable of asymptotically stabilizing reference trajectories which are reduced to a single point). Moreover, it has clear practical consequences because there are applications (automatic tracking of a human-driven car, for instance) for which the reference trajectory, and thus its properties, are not known in advance (is the leading car going to keep moving or stop?) so that one cannot easily decide on which controller to use. Switching between various controllers is a possible strategy, which has been studied by some authors and may give satisfactory results in many situations. However, since implementing a predefined switching strategy between two controllers sums up to designing a third controller, this does not solve the core of the problem nor grant any certitude of success.

A third issue, which is not specific to nonholonomic systems, but has seldom been addressed in the nonlinear control literature, concerns the problem of tracking *nonfeasible* trajectories (i. e., trajectories which are not solutions to the system's equations). Since exact tracking is not possible, by the definition of a nonfeasible trajectory, the control objective is then to ensure that the tracking errors shrink to, and ever after never exceed, certain nonzero thresholds. The fact that these thresholds can theoretically be arbitrarily small in the case of nonholonomic systems, if the amplitude of the velocity control inputs is not limited, makes this problem particularly relevant for these systems. This can be termed as a *practical* stabilization objective which, although slightly less ambitious than the objective of asymptotic stabilization considered in previous sections, opens up both the control design problem and the range of applications significantly. For instance, it allows to address the problem of tracking an omnidirectional vehicle with a unicycle-type, or a car-like, vehicle. In the context of planning a trajectory with obstacle avoidance, transforming a nonfeasible trajectory into a feasible approximation for a certain mobile robot can be performed by applying a practical stabilizer to a model of this robot and by numerical integration of the system's closed-loop equations. Also, if one reformulates the former question about the existence of a *universal* stabilizer, with the objective of asymptotic stabilization now replaced by the one of practical stabilization, then the answer becomes positive: such a stabilizer exists and, moreover, the reference trajectories do not even have to be feasible.

34.4.6 Practical Stabilization of Arbitrary Trajectories Based on the Transverse Function Approach

A possible approach for the design of practical stabilizers in the case of controllable driftless systems is described in [34.8]. Some of its basic principles, here adapted to the specific examples of unicycle-type and car-like mobile robots, are recalled next.

Let us introduce some matrix notation that will be used in this section.

$$R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}, \quad S = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

$$\bar{R}(\theta) = \begin{pmatrix} R(\theta) & 0 \\ 0 & 1 \end{pmatrix}.$$

Unicycle Case

With the above notation, the kinematic model (34.4) can be written as

$$\dot{g} = \bar{R}(\theta)Cu \quad (34.45)$$

with $g = (x, y, \theta)'$ and

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

Let us now consider a smooth function

$$f : \alpha \mapsto f(\alpha) = \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \\ f_\theta(\alpha) \end{pmatrix}$$

with $\alpha \in S^1 = \mathbb{R}/2\pi\mathbb{Z}$ (i. e., α is an angle variable), and define

$$\begin{aligned} \bar{g} &:= \begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{pmatrix} := g - \bar{R}(\theta - f_\theta(\alpha))f(\alpha) \\ &= \begin{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - R(\theta - f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ \theta - f_\theta(\alpha) \end{pmatrix}. \end{aligned} \quad (34.46)$$

Note that \bar{g} can be viewed as the situation of a frame $\bar{\mathcal{F}}_m(\alpha)$ the origin of which has components

$$-R(-f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix}$$

in the frame \mathcal{F}_m . In term of differential geometry, \bar{g} is the product of g by the inverse $f(\alpha)^{-1}$ of $f(\alpha)$, in the sense of the Lie group operations in $SE(2)$. Hence, $\bar{\mathcal{F}}_m(\alpha)$ is all the closer to \mathcal{F}_m as the components of $f(\alpha)$ are small. For any smooth time function $t \mapsto \alpha(t)$, and along any solution to system (34.45), the time derivative of \bar{g} is given by

$$\dot{\bar{g}} = \bar{R}(\bar{\theta})\bar{u} \quad (34.47)$$

with

$$\bar{u} = A(\alpha) \begin{pmatrix} \bar{R}(f_\theta(\alpha)) & -\partial f / \partial \alpha(\alpha) \end{pmatrix} \begin{pmatrix} Cu \\ \dot{\alpha} \end{pmatrix} \quad (34.48)$$

and

$$A(\alpha) = \begin{pmatrix} I_2 & -S \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix} \\ 0 & 1 \end{pmatrix}. \quad (34.49)$$

From (34.47) and (34.48), one can view $\dot{\alpha}$ as a complementary control input that can be used to monitor the motion of the frame $\tilde{\mathcal{F}}_m(\alpha)$. More precisely, $\tilde{\mathcal{F}}_m(\alpha)$ can be viewed as an omnidirectional frame provided that $\bar{\mathbf{u}}$ can be rendered equal to any vector of \mathbb{R}^3 , i. e., provided that the mapping $(\mathbf{u}, \dot{\alpha}) \mapsto \bar{\mathbf{u}}$ is onto. Let us determine when this condition is satisfied. Equation (34.48) can also be written as

$$\bar{\mathbf{u}} = A(\alpha)\mathbf{H}(\alpha) \begin{pmatrix} \mathbf{u} \\ \dot{\alpha} \end{pmatrix} \quad (34.50)$$

with

$$\mathbf{H}(\alpha) = \begin{pmatrix} \cos f_\theta(\alpha) & 0 & -\partial f_x / \partial \alpha(\alpha) \\ \sin f_\theta(\alpha) & 0 & -\partial f_y / \partial \alpha(\alpha) \\ 0 & 1 & -\partial f_\theta / \partial \alpha(\alpha) \end{pmatrix}. \quad (34.51)$$

Since $A(\alpha)$ is invertible, $\tilde{\mathcal{F}}_m(\alpha)$ is omnidirectional if and only if the matrix $\mathbf{H}(\alpha)$ is also invertible. A function f which satisfies this property for any $\alpha \in S^1$ is called a *transverse function* [34.9]. The issue of the existence of such functions has been treated in the much more general context of the transverse function approach [34.8, 9]. In the present case, a family of transverse functions is given by:

$$f(\alpha) = \begin{pmatrix} \varepsilon \sin \alpha \\ \varepsilon^2 \eta \frac{\sin 2\alpha}{4} \\ \arctan(\varepsilon \eta \cos \alpha) \end{pmatrix} \quad \text{with } \varepsilon, \eta > 0. \quad (34.52)$$

Indeed, with this function one can verify that, for any $\alpha \in S^1$, $\det \mathbf{H}(\alpha) = -\frac{\varepsilon^2 \eta}{2} \cos(\arctan(\varepsilon \eta \cos \alpha)) < 0$. Note that the components of f uniformly tend to zero as ε tends to zero, so that the associated omnidirectional frame $\tilde{\mathcal{F}}_m(\alpha)$ can be made arbitrarily close to \mathcal{F}_m by choosing ε small (but different from zero).

Now, let $t \mapsto \mathbf{g}_r(t) = [\mathbf{x}_r(t), \mathbf{y}_r(t), \theta_r(t)]^\top$ denote a smooth, but otherwise *arbitrary*, reference trajectory. It is not difficult to derive from (34.47) a feedback law $\bar{\mathbf{u}}$ which asymptotically stabilizes $\bar{\mathbf{g}}$ at \mathbf{g}_r . A possible choice is given by

$$\bar{\mathbf{u}} = \bar{\mathbf{R}}(-\bar{\theta})[\dot{\mathbf{g}}_r - k(\bar{\mathbf{g}} - \mathbf{g}_r)], \quad (34.53)$$

which implies that $(\dot{\bar{\mathbf{g}}} - \dot{\mathbf{g}}_r) = -k(\bar{\mathbf{g}} - \mathbf{g}_r)$ and therefore that $\bar{\mathbf{g}} - \mathbf{g}_r = 0$ is an exponentially stable equilibrium of the above equation for any $k > 0$. Then, it follows from (34.46) that

$$\lim_{t \rightarrow +\infty} \left\{ g(t) - g_r(t) - \bar{\mathbf{R}}[\theta_r(t)] f[\alpha(t)] \right\} = 0. \quad (34.54)$$

The norm of the tracking error $\|\mathbf{g} - \mathbf{g}_r\|$ is thus ultimately bounded by the norm of $f(\alpha)$ which, in view of (34.52), can be made arbitrarily small via the choice of ε . It is in this sense that practical stabilization is achieved. The control \mathbf{u} for the unicycle-like robot is then calculated by inverting the relation (34.50) and using the expression (34.53) of $\bar{\mathbf{u}}$.

While it can be tempting to use very small values of ε for the transverse function f in order to obtain a good tracking precision, one must be aware of the limits of this strategy. Indeed, when ε tends to zero, the matrix $\mathbf{H}(\alpha)$ defined by (34.51) becomes ill-conditioned, and its determinant tends to zero. This implies, by (34.50), that the robot's velocities u_1 and u_2 may become very large. In particular, when the reference trajectory \mathbf{g}_r is not feasible, many manoeuvres are likely to occur. Note that this difficulty is intrinsic to the robot's nonholonomy and that it cannot be circumvented (think about the problem of parking a car in a very narrow parking place). For this reason, trying to impose very accurate tracking of nonfeasible trajectories is not necessarily a good option in practice. On the other hand, when the trajectory is feasible, manoeuvres are not needed to achieve accurate tracking, so that smaller values of ε can be used in this case. This clearly leads to a dilemma when the reference trajectory is not known in advance and its properties in term of feasibility can vary with time. A control strategy which addresses this issue, based on the use of transverse functions whose magnitude can be adapted online, is proposed in [34.10]. Experimental validations of the present approach on a unicycle-like robot can also be found in [34.11].

Car Case

The control approach presented above can be extended to car-like vehicles (and also to the trailer case). Again, the idea is to associate with the robot's frame \mathcal{F}_m an omnidirectional *companion* frame $\tilde{\mathcal{F}}_m(\alpha)$ which can be maintained arbitrarily close to \mathcal{F}_m via the choice of some design parameters. Let us show how this can be done for a car-like vehicle. To simplify the forthcoming equations, let us rewrite system (34.5) as

$$\begin{cases} \dot{x} = u_1 \cos \theta, \\ \dot{y} = u_1 \sin \theta, \\ \dot{\theta} = u_1 \xi, \\ \dot{\xi} = u_\xi, \end{cases}$$

with $\xi = (\tan \phi)/L$ and $u_\xi = u_2(1 + \tan^2 \phi)/L$. This system can also be written as (compare with (34.45))

$$\begin{cases} \dot{\mathbf{g}} = \bar{\mathbf{R}}(\theta)C(\xi)u_1, \\ \dot{\xi} = u_\xi, \end{cases} \quad (34.55)$$

with $\mathbf{g} = (\mathbf{x}, \mathbf{y}, \theta)^\top$ and $C(\xi) = (1, 0, \xi)^\top$. Let us now consider a smooth function

$$f: \alpha \mapsto f(\alpha) = \begin{pmatrix} f_g(\alpha) \\ f_\xi(\alpha) \end{pmatrix} = \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \\ f_\theta(\alpha) \\ f_\xi(\alpha) \end{pmatrix}$$

with $\alpha \in S^1 \times S^1$ (i. e. $\alpha = (\alpha_1, \alpha_2)$), and define (compare with (34.46))

$$\begin{aligned} \bar{\mathbf{g}} &:= \begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{pmatrix} := \mathbf{g} - \bar{\mathbf{R}}(\theta - f_\theta(\alpha))f_g(\alpha) \\ &= \begin{pmatrix} x \\ y \\ \theta - f_\theta(\alpha) \end{pmatrix} - \mathbf{R}(\theta - f_\theta(\alpha)) \begin{pmatrix} f_x(\alpha) \\ f_y(\alpha) \end{pmatrix}, \end{aligned} \quad (34.56)$$

which, as in the unicycle case, can be viewed as the situation of some *companion* frame $\bar{\mathcal{F}}_m(\alpha)$. By differentiating $\bar{\mathbf{g}}$ along any smooth time function $t \mapsto \alpha(t)$ and any solution to system (34.55), one can verify that (34.47) is still satisfied, except that $\bar{\mathbf{u}}$ is now given by

$$\begin{aligned} \bar{\mathbf{u}} &= A(\alpha) \begin{pmatrix} \bar{\mathbf{R}}[f_\theta(\alpha)] & -\partial f_g/\partial \alpha_1(\alpha) & -\partial f_g/\partial \alpha_2(\alpha) \\ C(\xi)u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} \\ &\quad \times \begin{pmatrix} C(\xi)u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} \end{aligned} \quad (34.57)$$

rather than by (34.48) (with $A(\alpha)$ still defined by (34.49)). Using the fact that

$$\begin{aligned} C(\xi)u_1 &= C(f_\xi(\alpha))u_1 + \{C(\xi) - C[f_\xi(\alpha)]\}u_1 \\ &= \begin{pmatrix} 1 \\ 0 \\ f_\xi(\alpha) \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ \xi - f_\xi(\alpha) \end{pmatrix} u_1, \end{aligned}$$

(34.57) can also be written as

$$\bar{\mathbf{u}} = A(\alpha)\mathbf{H}(\alpha) \begin{pmatrix} u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix} + A(\alpha) \begin{pmatrix} 0 \\ 0 \\ u_1[\xi - f_\xi(\alpha)] \end{pmatrix} \quad (34.58)$$

with

$$\mathbf{H}(\alpha) = \begin{pmatrix} \cos f_\theta(\alpha) & -\partial f_x/\partial \alpha_1(\alpha) & -\partial f_x/\partial \alpha_2(\alpha) \\ \sin f_\theta(\alpha) & -\partial f_y/\partial \alpha_1(\alpha) & -\partial f_y/\partial \alpha_2(\alpha) \\ f_\xi(\alpha) & -\partial f_\theta/\partial \alpha_1(\alpha) & -\partial f_\theta/\partial \alpha_2(\alpha) \end{pmatrix}. \quad (34.59)$$

By setting

$$u_\xi = \dot{f}_\xi(\alpha) - k(\xi - f_\xi(\alpha)) \quad (34.60)$$

with $k > 0$, it follows from (34.55) that $\xi - f_\xi(\alpha)$ exponentially converges to zero. Hence, after some transient phase whose duration is commensurable with $1/k$, $\xi - f_\xi(\alpha) \approx 0$, and (34.58) reduces to

$$\bar{\mathbf{u}} = A(\alpha)\mathbf{H}(\alpha) \begin{pmatrix} u_1 \\ \dot{\alpha}_1 \\ \dot{\alpha}_2 \end{pmatrix}. \quad (34.61)$$

Provided that the function f is such that $\mathbf{H}(\alpha)$ is always invertible, this latter relation means that the frame $\bar{\mathcal{F}}_m(\alpha)$ associated with $\bar{\mathbf{g}}$ is omnidirectional. Any function f for which this property is satisfied is called a *transverse function*. Once it has been determined, one can proceed as in the unicycle case to asymptotically stabilize an arbitrary reference trajectory \mathbf{g}_r for $\bar{\mathbf{g}}$, for example, by defining $\bar{\mathbf{u}}$ as in (34.53). The control u_1 for the car is then obtained by inverting relation (34.61). The following lemma specifies a family of transverse functions for the car case.

Lemma 34.1

For any $\varepsilon > 0$ and any η_1, η_2, η_3 such that $\eta_1, \eta_2, \eta_3 > 0$ and $6\eta_2\eta_3 > 8\eta_3 + \eta_1\eta_2$, the function f defined by

$$f(\alpha) = \begin{pmatrix} \bar{f}_1(\alpha) \\ \bar{f}_4(\alpha) \\ \arctan(\bar{f}_3(\alpha)) \\ \bar{f}_2(\alpha)\cos^3 f_3(\alpha) \end{pmatrix}$$

with $\bar{f}: S^1 \times S^1 \rightarrow \mathbb{R}^4$ given by

$$\begin{aligned} \bar{f}(\alpha) &= \begin{pmatrix} \varepsilon(\sin \alpha_1 + \eta_2 \sin \alpha_2) \\ \varepsilon \eta_1 \cos \alpha_1 \\ \varepsilon^2 \left(\frac{\eta_1 \sin 2\alpha_1}{4} - \eta_3 \cos \alpha_2 \right) \\ \varepsilon^3 \left(\eta_1 \frac{\sin^2 \alpha_1 \cos \alpha_1}{6} - \frac{\eta_2 \eta_3 \sin 2\alpha_2}{4} - \eta_3 \sin \alpha_1 \cos \alpha_2 \right) \end{pmatrix} \end{aligned}$$

satisfies the transversality condition $\det \mathbf{H}(\alpha) \neq 0 \forall \alpha$, with $\mathbf{H}(\alpha)$ defined by (34.59) [34.12].

The simulation results reported in Fig. 34.12 illustrate the application of this control approach for a car-like robot. The reference trajectory is defined by the initial condition $g_r(0) = 0$ and its time derivative

$$\dot{g}_r(t) = \begin{cases} (0, 0, 0)^\top & \text{if } t \in [0, 30] \\ (1, 0, 0)^\top & \text{if } t \in [30, 38] \\ (0, 0.3, 0)^\top & \text{if } t \in [38, 53] \\ (-1, 0, 0)^\top & \text{if } t \in [53, 61] \\ (0, 0, 0.2)^\top & \text{if } t \in [61, 80] \end{cases}$$

This corresponds to a fixed situation when $t \in [0, 30]$, three sequences of pure translational motion when $t \in [30, 61]$, and a pure rotational motion when $t \in [61, 80]$. Let us remark that this trajectory is not feasible for the car-like robot when $t \in [38, 53]$, since it corresponds to a lateral translation in the direction of the unit vector j_r of the frame \mathcal{F}_r associated with g_r , nor when $t \in [61, 80]$, since a rear-drive car cannot perform pure rotational motion. The initial configuration of the car-like robot, at $t = 0$, is $g(0) = (0, 1.5, 0)$, and the initial steering wheel angle is $\phi(0) = 0$.

In Fig. 34.12a, the robot is drawn with a solid line at several time instants, whereas the chassis of the reference vehicle is shown as a dashed line at the same time instants. The figure also shows the trajectory of the point located at the mid-distance of the robot's rear wheels. Figure 34.12b shows the time evolution of the tracking error expressed in the reference frame (i. e., (x_e, y_e) as defined by (34.20), and $\theta_e = \theta - \theta_r$). It follows from (34.54) that, after the transient phase associated with the exponential convergence of \tilde{g} to zero, the ultimate bound for $|x_e|$, $|y_e|$, and $|\theta_e|$ is upper-bounded by the maximum amplitude of the functions f_x , f_y , and f_θ , respectively. For this simulation, the control parameters of the transverse function f of Lemma 34.1 have been chosen as follows: $\varepsilon = 0.17$, $\eta_{1,2,3} = (12, 2, 20)$. With these values, one can verify that $|f_x|$, $|f_y|$, and $|f_\theta|$ are bounded by 0.51, 0.11, and 0.6, respectively. This is consistent with the time evolution of the tracking error in

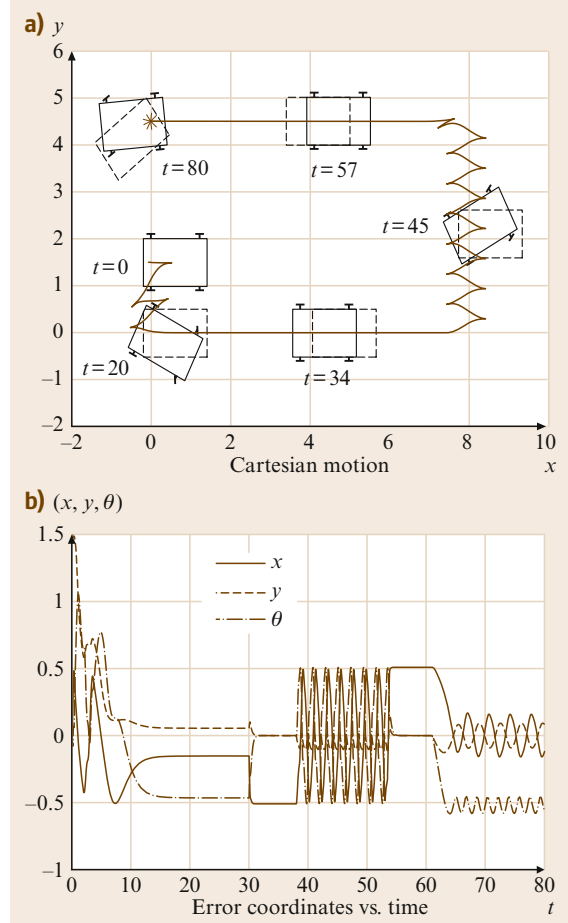


Fig. 34.12a,b Practical stabilization of an arbitrary trajectory by the transverse function approach

the figure. As pointed out for the unicycle case, a greater tracking precision could be obtained by decreasing the value of ε , but this would involve larger values of the control inputs and also more frequent manoeuvres, especially on the time intervals $[38, 53]$ and $[61, 80]$ when the reference trajectory is not feasible.

34.5 Complementary Issues and Bibliographical Guide

General Trailer Systems

Most of the control design approaches here presented and illustrated for unicycle-like and car-like vehicles can be extended to the case of trains of vehicles composed of trailers hitched to a leading vehicle. In particular, the

methods of Sect. 34.4 which are specific to nonholonomic systems can be extended to this case, provided that the kinematic equations of motion of the system can be transformed (at least semiglobally) into a chained system [34.5]. This basically requires that the hitch point

of each trailer is located on the rear-wheel axle of the preceding vehicle [34.13]. For instance, the transformation to the chained form is not possible when there are two (or more) successive trailers with off-axle hitch points [34.14]. So-called *general trailer systems* (with off-axle hitch points) raise difficult control design issues, and the literature devoted to them is sparse. For this reason, and also because these systems are not met in applications as frequently as simpler vehicles, control methods specifically developed for them are not reported here. Nonetheless, a few related references are given next. The path-following problem has been considered in, e.g., [34.15] for a system with two trailers and, more generally, in [34.16, Chap. 3] and [34.17] for an arbitrary number of trailers. To our knowledge, the problem of stabilizing non-stationary reference trajectories has not been addressed for these systems (except in the single trailer case for which the system can be transformed into the chained form [34.14, 18]). In fact, the explicit calculation of feasible trajectories joining a given configuration to another is already a very difficult problem, even in the absence of obstacles. As for the asymptotic stabilization of fixed configurations, the problem can (in theory) be solved by using existing general methods developed for the larger class of controllable driftless systems. However, the calculations associated with these methods quickly become intractable when the number of trailers increases. More specific and simpler ones have been proposed in [34.19], for an arbitrary number of trailers and the asymptotic stabilization of a reduced set of configurations, and in [34.20], in the case of two trailers and arbitrary fixed configurations.

Sensor-Based Motion Control

The control laws described in the present chapter, and their calculation, involve the online measurement, eventually complemented by the online estimation, of variables depending on the position of the vehicles in their environment. Measures can be acquired via the use of various sensors (odometry, GPS, proximetry, vision, . . .). Usually, various treatments are applied to raw sensory data prior to computing the control variables themselves. For instance, noise filtering and state estimation are such basic operations, well documented in the automatic-control literature. Among all sensors, vision sensors play a particularly important role in robotic applications, due to the richness and versatility of the information which they provide. The combination of visual data with feedback control is often referred to as *visual servoing*. In Chap. 24, a certain number of visual servoing tasks are addressed, mostly in the context of

manipulation and/or under the assumption that realizing the robot task is equivalent to controlling the pose of a camera mounted on an omnidirectional manipulator. In a certain number of cases, the concepts and methods described in this chapter can be adapted, without much effort, to the context of mobile robots. These cases basically correspond to the control methods adapted from robotic manipulation which are described in Sect. 34.3 of the present chapter. For instance, automatic driving via the control of the visually estimated lateral distance between a robotic vehicle and the side of a road, or car-platooning by controlling the frontal and lateral distances to a leading vehicle, can be addressed with the control techniques reported in Chap. 24. The reason is that it is possible to simply recast these techniques in the form of the control laws proposed in Sect. 34.3. However, there are also vision-based applications for nonholonomic mobile robots which cannot be solved by applying classical visual-servoing techniques. This is the case, for instance, of the task objectives addressed in Sect. 34.4, an example of which is the stabilization of the complete posture (i.e., position and orientation) of a nonholonomic vehicle at a desired one. Vision-based control problems of this type have been addressed in [34.10, 21].

A few former surveys on the control of WMRs have been published. Let us mention [34.22–24], which contain chapters on the modeling and control issues. A detailed classification of kinematic and dynamic models for the different types of WMR structures, on which Chap. 17 is based, is provided in [34.25]. The use of the chain form to represent WMR equations has been proposed in [34.26], then generalized in [34.13].

Path following may have been the first mobile robot control problem addressed by researchers in robotics. Among the pioneering works, let us cite [34.27, 28]. Several results presented in the present chapter are based on [34.5, 29].

The problem of tracking admissible trajectories for unicycle-type and car-like vehicles is treated in the books [34.22–24], and also in numerous conference and journal papers. Several authors have addressed this problem by applying dynamic feedback linearization techniques. In this respect, one can consult [34.30–32], and [34.22, Chap. 8], for instance.

Numerous papers on the asymptotic stabilization of fixed configurations have been published. Among them, [34.33] provides an early overview of feedback control techniques elaborated for this purpose, and also a list of references. The first result presenting a time-varying feedback solution to this problem, in the case

of a unicycle-type vehicle, is in [34.3]. The conference paper [34.34] provides a survey on time-varying feedback stabilization, in the more general context of nonlinear control systems. More specific results, like Propositions 34.9 and 34.12, are given in [34.1, 5]. Other early results on the design of smooth time-varying feedbacks can be found in [34.35, 36], for example. Concerning continuous (but not Lipschitz-continuous) time-varying feedbacks yielding exponential convergence, one can consult [34.37]. Designs of hybrid discrete/continuous fixed-point stabilizers can be found in [34.38–40], for instance. The control law in Proposition 34.14 is taken from [34.6]. Discontinuous control

design techniques are not addressed in the present chapter, but the interested reader will find examples of such feedbacks in [34.41, 42].

To our knowledge, the control approach presented in Sect. 34.4.6, which is based on the concept of transverse functions [34.8, 9], is the first attempt to address the problem of tracking *arbitrary* trajectories (i.e., not necessarily feasible for the controlled robot). Implementation issues and experimental results for this approach can be found in [34.10, 11]. An overview of trajectory tracking problems for wheeled mobile robots, with a detailed case study of car-like systems, is presented in [34.12].

References

- 34.1 P. Morin, C. Samson: Control of non-linear chained systems. From the Routh–Hurwitz stability criterion to time-varying exponential stabilizers, *IEEE Trans. Autom. Control* **45**, 141–146 (2000)
- 34.2 R.W. Brockett: Asymptotic stability and feedback stabilization. In: *Differential Geometric Control Theory*, ed. by R.W. Brockett, R.S. Millman, H.J. Sussmann (Birkhäuser, Boston 1983)
- 34.3 C. Samson: Velocity and torque feedback control of a nonholonomic cart, *Proc. Int. Workshop Adapt. Nonlinear Control: Issues in Robotics* (1990), also in *LNCIS*, Vol. 162, (Springer, Berlin, Heidelberg 1991)
- 34.4 J.-M. Coron: Global asymptotic stabilization for controllable systems without drift, *Math. Control Signals Syst.* **5**, 295–312 (1992)
- 34.5 C. Samson: Control of chained systems. Application to path following and time-varying point-stabilization, *IEEE Trans. Autom. Control* **40**, 64–77 (1995)
- 34.6 P. Morin, C. Samson: Exponential stabilization of nonlinear driftless systems with robustness to unmodeled dynamics, *ESAIM: Control Optim. Calc. Var.* **4**, 1–36 (1999)
- 34.7 D.A. Lizárraga: Obstructions to the existence of universal stabilizers for smooth control systems, *Math. Control Signals Syst.* **16**, 255–277 (2004)
- 34.8 P. Morin, C. Samson: Practical stabilization of driftless systems on Lie groups: the transverse function approach, *IEEE Trans. Autom. Control* **48**, 1496–1508 (2003)
- 34.9 P. Morin, C. Samson: A characterization of the Lie algebra rank condition by transverse periodic functions, *SIAM J. Control Optim.* **40**(4), 1227–1249 (2001)
- 34.10 G. Artus, P. Morin, C. Samson: Control of a maneuvering mobile robot by transverse functions, *Proc. Symp. Adv. Robot Kinemat. (ARK)* (2004) pp. 459–468
- 34.11 G. Artus, P. Morin, C. Samson: Tracking of an omnidirectional target with a nonholonomic mobile robot, *Proc. IEEE Conf. Adv. Robotics (ICAR)* (2003) pp. 1468–1473
- 34.12 P. Morin, C. Samson: Trajectory tracking for nonholonomic vehicles: overview and case study, *Proc. 4th Int. Workshop Robot Motion Control (RoMoCo)*, ed. by K. Kozłowski (2004) pp. 139–153
- 34.13 O.J. Sørdaalen: Conversion of the kinematics of a car with n trailers into a chained form, *Proc. IEEE Int. Conf. Robot. Autom.* (1993) pp. 382–387
- 34.14 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness, motion planning and trailer systems, *Proc. IEEE Int. Conf. Decis. Control* (1993) pp. 2700–2705
- 34.15 P. Bolzern, R.M. DeSantis, A. Locatelli, D. Masciocchi: Path-tracking for articulated vehicles with off-axle hitching, *IEEE Trans. Control Syst. Technol.* **6**, 515–523 (1998)
- 34.16 D.A. Lizárraga: Contributions à la stabilisation des systèmes non-linéaires et à la commande de véhicules sur roues. Ph.D. Thesis (INRIA-INPG, University of Grenoble, France 2000)
- 34.17 C. Altafini: Path following with reduced off-tracking for multibody wheeled vehicles, *IEEE Trans. Control Syst. Technol.* **11**, 598–605 (2003)
- 34.18 F. Lamiroux, J.-P. Laumond: A practical approach to feedback control for a mobile robot with trailer, *Proc. IEEE Int. Conf. Robotics Autom.* (1998) pp. 3291–3296
- 34.19 D.A. Lizárraga, P. Morin, C. Samson: Chained form approximation of a driftless system. Application to the exponential stabilization of the general N -trailer system, *Int. J. Control* **74**, 1612–1629 (2001)
- 34.20 M. Venditelli, G. Oriolo: Stabilization of the general two-trailer system, *Proc. IEEE Int. Conf. Robot. Autom.* (2000) pp. 1817–1823
- 34.21 M. Maya-Mendez, P. Morin, C. Samson: Control of a nonholonomic mobile robot via sensor-based tar-

- get tracking and pose estimation, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2006) pp.5612–5618
- 34.22 C. Canudas de Wit, B. Siciliano, G. Bastin: *Theory of Robot Control* (Springer, Berlin, Heidelberg 1996)
- 34.23 J.-P. Laumond (Ed.): *Robot Motion Planning and Control*, Lect. Notes Control Inform. Sci., Vol. 229 (Springer, Berlin, Heidelberg 1998)
- 34.24 Y.F. Zheng (Ed.): *Recent Trends in Mobile Robots*, World Scientific Ser. Robotics Automat. Syst., Vol. 11 (World Scientific, Singapore 1993)
- 34.25 G. Campion, G. Bastin, B. d'Andréa-Novel: Structural properties and classification of kinematic and dynamic models of wheeled mobile robots, IEEE Trans. Robot. Autom. **12**, 47–62 (1996)
- 34.26 R.M. Murray, S.S. Sastry: Steering nonholonomic systems in chained form, Proc. IEEE Int. Conf. Decis. Control (1991) pp.1121–1126
- 34.27 E.D. Dickmanns, A. Zapp: Autonomous high speed road vehicle guidance by computer vision, Proc. IFAC World Congr. (1987)
- 34.28 W.L. Nelson, I.J. Cox: Local path control for an autonomous vehicle, Proc. IEEE Int. Conf. Robot. Autom. (1988) pp.1504–1510
- 34.29 C. Samson: Path following and time-varying feedback stabilization of a wheeled mobile robot, Proc. Int. Conf. Autom. Robot. Comput. Vis. (1992)
- 34.30 B. d'Andréa-Novel, G. Campion, G. Bastin: Control of nonholonomic wheeled mobile robots by state feedback linearization, Int. J. Robot. Res. **14**, 543–559 (1995)
- 34.31 A. De Luca, M.D. Di Benedetto: Control of nonholonomic systems via dynamic compensation, Kybernetika **29**, 593–608 (1993)
- 34.32 M. Fliess, J. Lévine, P. Martin, P. Rouchon: Flatness and defect of non-linear systems: introductory theory and examples, Int. J. Control **61**, 1327–1361 (1995)
- 34.33 I. Kolmanovsky, N.H. McClamroch: Developments in nonholonomic control problems, IEEE Control Syst. **15**, 20–36 (1995)
- 34.34 P. Morin, J.-B. Pomet, C. Samson: Developments in time-varying feedback stabilization of nonlinear systems, Proc. IFAC NOLCOS (1998) pp. 587–594
- 34.35 J.-B. Pomet: Explicit design of time-varying stabilizing control laws for a class of controllable systems without drift, Syst. Control Lett. **18**, 467–473 (1992)
- 34.36 A.R. Teel, R.M. Murray, G. Walsh: Nonholonomic control systems: from steering to stabilization with sinusoids, Int. J. Control **62**, 849–870 (1995), also in Proc. IEEE Int. Conf. Decis. Control (1992) pp. 1603–1609
- 34.37 R.T. M'Closkey, R.M. Murray: Exponential stabilization of driftless nonlinear control systems using homogeneous feedback, IEEE Trans. Autom. Control **42**, 614–6128 (1997)
- 34.38 M.K. Bennani, P. Rouchon: Robust stabilization of flat and chained systems, Proc. Eur. Control Conf. (1995) pp.2642–2646
- 34.39 P. Lucibello, G. Oriolo: Stabilization via iterative state feedback with application to chained-form systems, Proc. IEEE Conf. Decis. Control (1996) pp.2614–2619
- 34.40 O.J. Sørдалen, O. Egeland: Exponential stabilization of nonholonomic chained systems, IEEE Trans. Autom. Control **40**, 35–49 (1995)
- 34.41 A. Astolfi: Discontinuous control of nonholonomic systems, Syst. Control Lett. **27**, 37–45 (1996)
- 34.42 C. Canudas de Wit, O.J. Sørдалen: Exponential stabilization of mobile robots with nonholonomic constraints, IEEE Trans. Autom. Control **37**(11), 1791–1797 (1992)

35. Motion Planning and Obstacle Avoidance

Javier Minguez, Florent Lamiroux, Jean-Paul Laumond

This chapter describes motion planning and obstacle avoidance for mobile robots. We will see how both areas do not share the same modeling background. From the very beginning of motion planning, the research has been dominated by computer science. Researchers aim at devising well-grounded algorithms with well-understood completeness and exactness properties. The introduction of nonholonomic constraints has forced these algorithms to be revisited via the introduction of differential geometry approaches. Such a combination has been made possible for certain classes of systems, so-called small-time controllable ones. The underlying hypothesis of motion planning algorithms remains the knowledge of a global and accurate map of the environment. More than that, the considered system is a formal system of equations that does not account for the entire physical system: uncertainties in the world or system modeling are not considered. Such hypotheses are too strong in practice. This is why other complementary researchers have adopted a parallel, more pragmatic but realistic approach to deal with obstacle avoidance. The problem here is not to deal with complicated systems like a car with multiple trailers. The considered systems are much simpler with respect to their geometric shape. The problem considers sensor-based motion to face the physical issues of a real system navigating in a real world better than motion planning algorithms: how to *navigate* toward a goal in a cluttered environment when the obstacles to avoid are discovered in real time? This is the question obstacle avoidance addresses.

35.1 **Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory ...** 828

35.2	Kinematic Constraints and Controllability	829
35.2.1	Definitions	829
35.2.2	Controllability	829
35.2.3	Example: The Differentially Driven Mobile Robot	830
35.3	Motion Planning and Small-Time Controllability	830
35.3.1	The Decision Problem	830
35.3.2	The Complete Problem	831
35.4	Local Steering Methods and Small-Time Controllability	832
35.4.1	Local Steering Methods that Account for Small-Time Controllability	832
35.4.2	Equivalence Between Chained-Formed and Feedback-Linearizable Systems	834
35.5	Robots and Trailers	835
35.5.1	Differentially Driven Mobile Robots	835
35.5.2	Differentially Driven Mobile Robots Towing One Trailer	835
35.5.3	Car-Like Mobile Robots	835
35.5.4	Bi-steerable Mobile Robots	835
35.5.5	Differentially Driven Mobile Robots Towing Trailers	836
35.5.6	Open Problems	836
35.6	Approximate Methods	837
35.6.1	Forward Dynamic Programming	837
35.6.2	Discretization of the Input Space ...	837
35.6.3	Input-Based Rapidly Exploring Random Trees	837
35.7	From Motion Planning to Obstacle Avoidance	837
35.8	Definition of Obstacle Avoidance	838
35.9	Obstacle Avoidance Techniques	839
35.9.1	Potential Field Methods (PFM)	839
35.9.2	Vector Field Histogram (VFH)	840

35.9.3 The Obstacle Restriction Method (ORM).....	841
35.9.4 Dynamic Window Approach (DWA) .	842
35.9.5 Velocity Obstacles (VO).....	843
35.9.6 Nearness Diagram Navigation (ND)	843
35.10 Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance	845
35.10.1 Techniques that Abstract the Vehicle Aspects...	845
35.10.2 Techniques of Decomposition in Subproblems .	846
35.11 Integration Planning – Reaction.....	847
35.11.1 Path Deformation Systems.....	847
35.11.2 Tactical Planning Systems.....	848
35.12 Conclusions, Future Directions, and Further Reading.....	849
References	850

The appearance of mobile robots in the late 1960s to early 1970s initiated a new research domain: autonomous navigation. It is interesting to notice that the first navigation systems were published in the very first International Joint Conferences on Artificial Intelligence (IJCAI 1969). These systems were based on seminal ideas which have been very fruitful in the development of robot motion planning algorithms. For instance, in 1969, the mobile robot Shakey used a grid-based approach to model and explore the environment [35.1]; in 1977 Jason used a visibility graph built from the corners of the obstacles [35.2]; in 1979 Hilare decomposed the environment into collision-free convex cells [35.3].

In the late 1970s, studies of robot manipulators popularized the notion of configuration space of a mechanical system [35.4]. In the configuration space the *piano* becomes a point. The motion planning problem for a mechanical system was thus reduced to finding a path for a point in the configuration space. The way was open to extend the seminal ideas and to develop new and well-grounded algorithms (see Latombe’s book [35.5]).

One decade later, the notion of nonholonomic systems (also borrowed from mechanics) appeared in the

literature [35.6] on robot motion planning through the problem of car parking. This problem had not been solved by the pioneering works on mobile robot navigation. Nonholonomic motion planning then became an attractive research field [35.7].

Besides this research effort in path planning, work was initiated in order to make robots move out of their initially artificial environments where the world was cylindrical and composed of wooden vertical boards. Robots started to move in laboratory buildings, with people walking around. Inaccurate localization, uncertain and incomplete maps of the world, and unexpected moving or static obstacles made roboticists aware of the gap between planning a path and executing a motion. From then on, the domain of obstacle avoidance has been very active.

The challenge of this chapter is to present both nonholonomic motion planning (Sects. 35.1–35.6) and obstacle avoidance (Sects. 35.7–35.10) issues. Section 35.11 reviews recent successful approaches that tend to embrace the whole problem of motion planning and motion control. These approaches benefit from both nonholonomic motion planning and obstacle avoidance methods.

35.1 Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory

Nonholonomic constraints are nonintegrable linear constraints over the velocity space of a system. For instance, the rolling without slipping constraint of a differentially driven mobile robot (Fig. 35.1) is linear with respect to the velocity vector (vector of linear and angular velocities) of the differentially driven robot and is nonintegrable (it cannot be integrated into a constraint over the configura-

tion variables). As a consequence, a differentially driven mobile robot can go anywhere but not following any trajectory. Other types of nonholonomic constraints arise when considering second-order differential equations such as the conservation of inertial momentum. A number of papers investigate the famous falling cat problem or free-floating robots in space (see [35.7] for an overview). This chapter is devoted

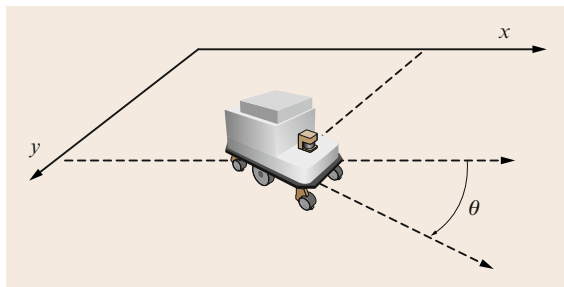


Fig. 35.1 A differentially driven mobile robot is subject to one linear kinematic constraint, due to the rolling without slipping constraint of the wheel axis

to nonholonomic constraints for mobile robots with wheels.

While the constraints due to obstacles are expressed directly in the configuration space, that is a manifold, nonholonomic constraints are expressed in the tangent space. In the presence of a linear kinematic constraint, the first question that naturally arises is: does this constraint reduce the space *reachable* by the system? This question can be answered by studying the structure of

the distribution spanned by the Lie algebra of the control system.

Even in the absence of obstacles, planning admissible motions (i. e., that satisfy the kinematic constraints) for a nonholonomic system between two configurations is not an easy task. Exact solutions have been proposed only for some classes of systems but a lot of systems remain without exact solution. In the general case however, an approximate solution can be used.

The motion planning problem for a nonholonomic system can be stated as follows: given a map of the environment with obstacles in the workspace, a robot subject to nonholonomic constraints, an initial configuration, and a goal configuration, find an admissible collision-free path between the initial and goal configurations. Solving this problem requires to take into account both the configuration space constraints due to obstacles and the nonholonomic constraints. The tools developed to address this issue thus combine motion planning and control theory techniques. Such a combination is possible for the class of so-called small-time controllable systems thanks to topological arguments (Theorem 35.2 in Sect. 35.3).

35.2 Kinematic Constraints and Controllability

In this section, we give the main definition about controllability, using *Sussman's* terminology [35.8].

35.2.1 Definitions

Let us denote by \mathcal{CS} the configuration space of dimension n of a given mobile robot and by \mathbf{q} the configuration of this robot. If the robot is mounted on wheels, it is subject to kinematic constraints, linear in the velocity vector:

$$\omega_i(\mathbf{q})\dot{\mathbf{q}} = 0, \quad i \in \{1, \dots, k\}.$$

We assume that these constraints are linearly independent for any \mathbf{q} . Equivalently, for each \mathbf{q} , there exist $m = n - k$ linearly independent vectors $f_1(\mathbf{q}), \dots, f_m(\mathbf{q})$ such that the above constraints are equivalent to

$$\exists(u_1, \dots, u_m) \in \mathbb{R}^m, \quad \dot{\mathbf{q}} = \sum_{i=1}^m u_i f_i(\mathbf{q}). \quad (35.1)$$

Let us notice that the choice of vectors $f_i(\mathbf{q})$ is not unique. Fortunately, all the following developments are

valid whatever the choice we make. Moreover, if the linear constraints are smooth, vector fields f_1, \dots, f_m can be chosen smooth with respect to \mathbf{q} . We assume this condition from now on.

Let us define \mathcal{U} , a compact subset of \mathbb{R}^m . We denote by Σ the control system defined by (35.1), with $(u_1, \dots, u_m) \in \mathcal{U}$

Definition 35.1

Local and small-time controllability

1. Σ is *locally controllable about configuration \mathbf{q}* iff the set of configurations reachable from \mathbf{q} by an admissible trajectory contains a neighborhood of \mathbf{q} ,
2. Σ is *small-time controllable about configuration \mathbf{q}* iff the set of configurations reachable from \mathbf{q} by an admissible trajectory in time less than T contains a neighborhood of \mathbf{q} for any T ,

f_1, \dots, f_m are called *control vector fields* of Σ . A system that is small-time controllable about each configuration is said to be *small-time controllable*.

35.2.2 Controllability

Checking the controllability properties of a system requires the analysis of the control Lie algebra associated with the system. Let us illustrate what is the Lie bracket of two vector fields in an informal way. Consider two basic motions: *go along a straight line* and *turn on the spot*, supported by vector fields denoted by f and g , respectively. Now consider the following combination: go forward during a time t , turn clockwise during the same time t , go backward during the same time t , and then turn counterclockwise during the time t . The system reaches a configuration which is not the starting one. Of course when t tends to 0 the goal configuration is very close to the starting one. The direction indicated by such goal configurations when t tends to 0 correspond to a new vector field which is the Lie bracket of f and g . In a more mathematical formulation, the Lie bracket $[f, g]$ of two vector fields f and g is defined as being the vector field $\partial fg - \partial gf$. The k -th coordinate of $[f, g]$ is

$$[f, g][k] = \sum_{i=1}^n (g[i] \frac{\partial}{\partial x_i} f[k] - f[i] \frac{\partial}{\partial x_i} g[k]).$$

The following theorem [35.9] gives a powerful result for symmetric systems (a system is said to be symmetric when \mathcal{U} is symmetric with respect to the origin)

Theorem 35.1

A symmetric system is small-time controllable about configuration \mathbf{q} iff the rank of the vector space spanned by the family of vector fields f_i together with all their brackets is n at \mathbf{q} .

Checking the Lie algebra rank condition (LARC) on a control system consists of trying to build a basis of the tangent space from a basis (e.g., a P. Hall family) of the free Lie algebra spanned by the control vector fields. An algorithm is proposed in [35.10, 11].

35.2.3 Example: The Differentially Driven Mobile Robot

To illustrate the notions developed in this section, we consider the differentially driven mobile robot displayed in Fig. 35.1. The configuration space of this robot is $\mathbb{R}^2 \times \mathcal{S}^1$, and a configuration can be represented by $\mathbf{q} = (x, y, \theta)$, where (x, y) is the position in the horizontal plane of the center of the wheel axis of the robot and θ the orientation with respect to the x -axis. The rolling without slipping kinematic constraint:

$$-\dot{x} \sin \theta + \dot{y} \cos \theta = 0$$

is linear with respect to the velocity vector $(\dot{x}, \dot{y}, \dot{\theta})$. Therefore, the subspace of admissible velocities is spanned by two vector fields, for instance:

$$f_1(\mathbf{q}) = \begin{pmatrix} \cos \theta \\ \sin \theta \\ 0 \end{pmatrix} \quad \text{and} \quad f_2(\mathbf{q}) = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}. \quad (35.2)$$

The Lie bracket of these two vector fields is

$$f_3(\mathbf{q}) = \begin{pmatrix} \sin \theta \\ -\cos \theta \\ 0 \end{pmatrix}.$$

This implies that, about any configuration \mathbf{q} , the rank of the vector space spanned by $f_1(\mathbf{q})$, $f_2(\mathbf{q})$, $f_3(\mathbf{q})$ is 3 and therefore that the differentially driven mobile robot is small-time controllable.

35.3 Motion Planning and Small-Time Controllability

Motion planning raises two problems: the first one addresses the *existence* of a collision-free admissible path (the decision problem) while the second one addresses the *computation* of such a path (the complete problem).

35.3.1 The Decision Problem

From now on, we assume that the set of collision-free configurations is an open subset. This implies that contact configurations are assumed in collision.

Theorem 35.2

The existence of a collision-free admissible path for a symmetric small-time controllable mobile robot between two configurations is equivalent to the existence of a collision-free (not necessarily admissible) path between these configurations.

Proof: Let us consider a not necessarily admissible, collision-free path between two configurations \mathbf{q}_1 and \mathbf{q}_2 as a continuous mapping Γ from interval $[0, 1]$ into the configuration space \mathcal{CS} such that:

1. $\Gamma(0) = \mathbf{q}_1, \Gamma(1) = \mathbf{q}_2$,
2. for any $t \in [0, 1]$, $\Gamma(t)$ is collision free.

Point 2 implies that, for any t , there exists a neighborhood $U(t)$ of $\Gamma(t)$ included in the collision-free subset of the configuration space.

Let us denote by $\varepsilon(t)$ the bigger lower bound of the time to collision of all the trajectories starting from $\Gamma(t)$. As the control vector (u_1, \dots, u_m) remains in the compact set \mathcal{U} , $\varepsilon(t) > 0$.

As the system is small-time controllable about $\Gamma(t)$, the set reachable from $\Gamma(t)$ in time less than $\varepsilon(t)$ is a neighborhood of $\Gamma(t)$ that we denote by $V(t)$.

The collection $\{V(t), t \in [0, 1]\}$ is an open covering of the compact set $\{\Gamma(t), t \in [0, 1]\}$. Therefore, we can extract a finite covering: $\{V(t_1), \dots, V(t_l)\}$, where $t_1 = 0 < t_2 < \dots < t_{l-1} < t_l = 1$ such that, for any i between 1 and $l-1$, $V(t_i) \cap V(t_{i+1}) \neq \emptyset$. For each i between 1 and $l-1$, we choose one configuration \mathbf{r}_i in $V(t_i) \cap V(t_{i+1})$. As the system is symmetric, there exists an admissible collision-free path between $\mathbf{q}(t_i)$ and \mathbf{r}_i and between \mathbf{r}_i and $\mathbf{q}(t_{i+1})$. The concatenation of these paths is a collision-free admissible path between \mathbf{q}_1 and \mathbf{q}_2 . ■

35.3.2 The Complete Problem

In the former section, we established that the decision problem, i.e., determining whether there exists a collision-free admissible path between two configurations, is equivalent to determining whether the configurations lie in the same connected component of the collision-free configuration space. In this section we present the tools necessary to solve the complete problem. These tools blend ideas from the classical motion planning problem addressed in Chap. 5 and from open-loop control theory, but require specific developments that we are going to present in the next section. Two main approaches have been devised in order to plan admissible collision-free motions for nonholonomic systems. The first one, proposed by [35.12], exploits the idea of the proof of Theorem 35.2 by recursively approximating a not necessarily admissible, collision-free path by a sequence of feasible paths. The second approach replaces the local method of probabilistic roadmap method (PRM) algorithms (Chap. 5) by a local steering method that connects configuration pairs by admissible paths.

Both approaches use a steering method. Before briefly describing them, we give the definition of a local steering method.

Definition 35.2

A local steering method for system Σ is a mapping

$$\begin{aligned} \mathcal{S}_{\text{loc}} : \mathcal{CS} \times \mathcal{CS} &\rightarrow C_{\text{pw}}^1([0, 1], \mathcal{CS}), \\ (\mathbf{q}_1, \mathbf{q}_2) &\mapsto \mathcal{S}_{\text{loc}}(\mathbf{q}_1, \mathbf{q}_2), \end{aligned}$$

where $\mathcal{S}_{\text{loc}}(\mathbf{q}_1, \mathbf{q}_2)$ is a piecewise continuously differentiable curve in \mathcal{CS} satisfying the following properties:

1. $\mathcal{S}_{\text{loc}}(\mathbf{q}_1, \mathbf{q}_2)$ satisfies the kinematic constraints associated to Σ ,
2. $\mathcal{S}_{\text{loc}}(\mathbf{q}_1, \mathbf{q}_2)$ connects \mathbf{q}_1 to \mathbf{q}_2 : $\mathcal{S}_{\text{loc}}(\mathbf{q}_1, \mathbf{q}_2)(0) = \mathbf{q}_1$, $\mathcal{S}_{\text{loc}}(\mathbf{q}_1, \mathbf{q}_2)(1) = \mathbf{q}_2$.

Approximation of a Not Necessarily Admissible Path

A not necessarily admissible collision-free path $\Gamma(t)$, $t \in [0, 1]$ connecting two configurations and a local steering method \mathcal{S}_{loc} being given, the approximation algorithm proceeds recursively by calling the function approximation defined by Algorithm 35.1 with inputs Γ , 0, and 1.

Algorithm 35.1

```

approximation function: inputs are a path  $\Gamma$  and
two abscissas  $t_1$  and  $t_2$  along this path
if  $\mathcal{S}_{\text{loc}}(\Gamma(t_1), \Gamma(t_2))$  collision free then
    return  $\mathcal{S}_{\text{loc}}(\Gamma(t_1), \Gamma(t_2))$ 
else
    return concat(approximation( $\Gamma, t_1, (t_1+t_2)/2$ ),
    approximation( $\Gamma, (t_1+t_2)/2, t_2$ ))
endif

```

Sampling-Based Roadmap Methods

Most sampling-based roadmap methods as described in Chap. 5 can be adapted to nonholonomic systems by replacing the connection method between pairs of configurations by a local steering method. This strategy is rather efficient for PRM algorithms. For the rapid random trees (RRT) method, the efficiency strongly depends on the metric used to choose the nearest neighbor. The distance function between two configurations needs to account for the length of the path returned by the local steering method to connect these configurations [35.13].

35.4 Local Steering Methods and Small-Time Controllability

The approximation algorithm described in the former section is recursive and raises the completeness question: does the algorithm finish in finite time or may it fail to find a solution?

A sufficient condition for the approximation algorithm to find a solution in a finite number of iterations is that the local steering method accounts for small-time controllability.

Definition 35.3

A local steering method \mathcal{S}_{loc} accounts for the small-time controllability of system Σ iff

For any $q \in \mathcal{CS}$, for any neighborhood U of q , there exists a neighborhood V of q such that for any $r \in V$, $\mathcal{S}_{\text{loc}}(q, r)([0, 1]) \subset U$

In other words, a local steering method accounts for the small-time controllability of a system if it produces paths getting closer to the configurations it connects when these configurations get closer to each other.

This property is also sufficient for probabilistic completeness of roadmap sampling-based methods.

35.4.1 Local Steering Methods that Account for Small-Time Controllability

Constructing a local steering method that accounts for small-time controllability is a difficult task that has been achieved only for a few classes of systems. Most mobile robots studied in the domain of motion planning are wheeled mobile robots, towing trailers or not.

Steering Using Optimal Control

The simplest system, namely the differentially driven robot presented in Sect. 35.2.3 with bounded velocities or the so-called *Reeds and Shepp* (RS) [35.14] car with bounded curvature have the same control vector fields (35.2). The difference lies in the domain of the control variables:

- $-1 \leq u_1 \leq 1, |u_2| \leq |u_1|$ for the RS car,
- $|u_1| + b|u_2| \leq 1$ for the differentially driven robot

where b is half of the distance between the right and left wheels. For these systems, a local steering method accounting for small-time controllability can be constructed using optimal control theory. For any admissible path defined over an interval I of one of these systems, we define a length as follows:

$$\int_I |u_1| \quad \text{for RS car ,}$$

$$\int_I |u_1| + b|u_2| \quad \text{for the differentially driven robot .}$$

The length of the shortest path between two configurations defines in both cases a metric over the configuration space. The synthesis of the shortest paths, i.e., the determination of the shortest path between any pair of configurations has been achieved, respectively, by [35.15] for RS car and later by [35.16] for the differentially driven (DD) robot. Figure 35.2 shows a representation of the balls corresponding to these metrics.

Optimal control naturally defines a local steering method that associates to any pair of configurations,

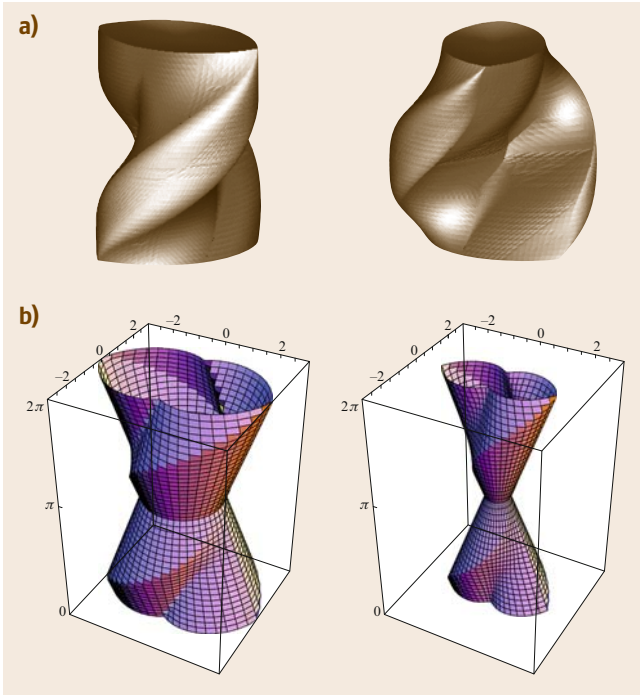


Fig. 35.2 (a) Two perspective views of an RS ball: the set of configurations reachable by a path of length less than a given distance for Reeds and Shepp car. (b) Perspective views of two DD balls: the set of configurations reachable by a path of length less than a given distance for the differentially driven robot. The orientation θ is represented on the z -axis

a shortest path between these configurations. Let us notice that the shortest path is unique between most pairs of configurations. A general result states that the collection of balls of radius $r > 0$ centered about \mathbf{q} induced by nonholonomic metrics constitute an increasing collection of neighborhoods of \mathbf{q} the intersection of which is $\{\mathbf{q}\}$. This property directly implies that local steering methods based on shortest paths account for small-time controllability.

The main advantage of optimal control is that it provides both a local steering method and a distance metric consistent with the steering method. This makes the steering methods well suited for path planning algorithms designed for holonomic systems and using a distance function, such as RRT (Chap. 5) for instance.

Unfortunately, the synthesis of the shortest paths has been realized only for the two simple systems described in this section. For more complex systems, the problem remains open.

The main drawback of the shortest path based steering methods described in this section is that input functions are not continuous. This requires an additional step to compute a time parameterization of the paths before motion execution. Along this time parameterization, input discontinuities force the robot to stop, for instance, to follow two successive arcs of circles of opposite curvature a mobile robot needs to stop between the arcs of circle in order to ensure continuity of the linear and angular velocities u_1 and u_2 .

Steering Chained-Form Systems

Some classes of systems can be put into what is called a *chained-form* system by a change of variable:

$$\dot{z}_1 = u_1, \quad (35.3)$$

$$\dot{z}_2 = u_2, \quad (35.4)$$

$$\dot{z}_3 = z_2 u_1, \quad (35.5)$$

$$\vdots \quad (35.6)$$

$$\dot{z}_n = z_{n-1} u_1. \quad (35.7)$$

Let us consider the following inputs [35.17]:

$$\begin{cases} u_1(t) = a_0 + a_1 \sin \omega t, \\ u_2(t) = b_0 + b_1 \cos \omega t + \dots + b_{n-2} \cos(n-2)\omega t. \end{cases} \quad (35.8)$$

Let $Z^{\text{start}} \in \mathbb{R}^n$ be a starting configuration. Each $z_i(1)$ can be computed from the coordinates of Z^{start} and parameters $(a_0, a_1, b_0, b_1, \dots, b_{n-2})$. For a given $a_1 \neq 0$ and a given configuration Z^{start} , the mapping from

$(a_0, b_0, b_1, b_2, b_3)$ to $Z(1)$ is a C^1 -diffeomorphism at the origin; the system is then invertible. For n smaller or equal to 5, the parameters $(a_0, b_0, b_1, \dots, b_{n-2})$ can be analytically computed from the coordinates of the two configurations Z^{start} and Z^{goal} . The corresponding sinusoidal inputs steer the system from Z^{start} to Z^{goal} . The shape of the path only depends on the parameter a_1 . Each value of a_1 thus defines a local steering method, denoted by $\mathcal{S}_{\text{sin}}^{a_1}$. None of these steering methods account for small-time controllability since, for any $Z \in \mathbb{R}^n$, $\mathcal{S}_{\text{sin}}^{a_1}(Z, Z)([0, 1])$ is not reduced to $\{Z\}$. To construct a local steering method accounting for small-time controllability from the collection of $\mathcal{S}_{\text{sin}}^{a_1}$, we need to make a_1 depend on the configurations Z_1 and Z_2 that we want to connect:

$$\lim_{Z^2 \rightarrow Z^1} a_1(Z^1, Z^2) = 0,$$

$$\lim_{Z^2 \rightarrow Z^1} a_0(Z^1, Z^2, a_1(Z^1, Z^2)) = 0,$$

$$\lim_{Z^2 \rightarrow Z^1} b_i(Z^1, Z^2, a_1(Z^1, Z^2)) = 0.$$

Such a construction is achieved in [35.18].

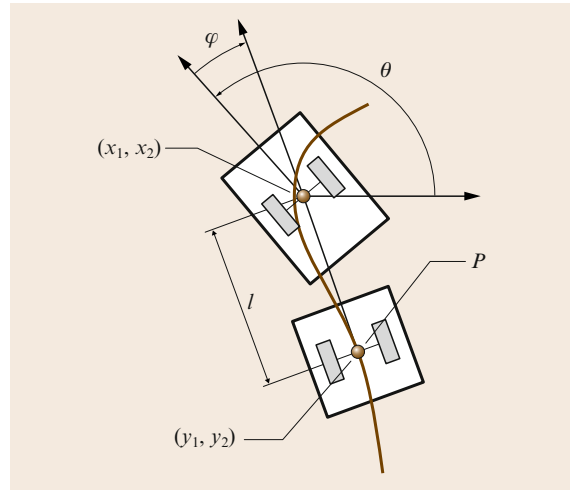


Fig. 35.3 A differentially driven robot towing a trailer hitched on top of the wheel axis of the robot is a feedback-linearizable system. The linearizing output is the center of the wheel axis of the trailer. The configuration of the system can be reconstructed by differentiating the curve $\mathbf{y}(s)$, where s is an arc-length parameterization, followed by linearizing the output. The orientation of the trailer is given by $\tau = \arctan(\dot{y}_2/\dot{y}_1)$. The angle between the robot and the trailer is given by $\varphi = -l \arctan(d\tau/ds)$, where l is the length of the trailer connection

Steering Feedback-Linearizable Systems

The concept of *feedback linearizability* (or differential flatness) was introduced by *Fliess et al.* [35.19, 20].

A system is said to be *feedback linearizable* if there exists an output (i.e., a function of the state, input, and input derivatives) called the *linearizing output*, such that the state and the input of the system are functions of the linearizing output and its derivatives. The dimension of the linearizing output is the same as the dimension of the input.

Let us illustrate this notion with a simple example. We consider a differentially driven mobile robot towing a trailer hitched on top of the wheel axis of the robot, as displayed in Fig. 35.3. The tangent to the curve followed by the center of the wheel axis of the trailer gives the orientation of the trailer. From the orientation of the trailer along the curve, we can deduce the curve followed by the center of the robot. The tangent to the curve followed by the center of the robot gives the orientation of the robot. Thus the linearizing output of this system is the center of the wheel axis of the trailer. By differentiating the linearizing output twice, we can reconstruct the configuration of the system.

Feedback linearizability is very interesting for steering purposes. Indeed, the linearizing output is not subject to any kinematic constraints. Therefore, if we know the relation between the state and the linearizing output, planning an admissible path between two configurations simply consists of building a curve in \mathbb{R}^m , where m is the dimension of the input with differential constraints at both ends. This problem can be easily solved using, for instance, polynomials.

For two input driftless systems like Σ , the linearizing output only depends on state \mathbf{q} , and the state \mathbf{q} depends on the linearizing output through the parameterization invariant values, namely, the linearizing output \mathbf{y} , the orientation τ of the vector tangent to the curve followed by \mathbf{y} , and the successive derivatives of τ with respect to the curvilinear abscissa s .

Thus, the configuration of a two-input feedback-linearizable driftless system of dimension n can be represented by a vector $(\mathbf{y}, \tau, \tau^1, \dots, \tau^{n-3})$ representing the geometric properties of the curve followed by the linearizing output along an admissible path passing through the configuration.

Therefore, designing a local steering method for such a system is equivalent to associating to any pair of vectors $(\mathbf{y}_1, \tau_1, \tau_1^1, \dots, \tau_1^{n-3}), (\mathbf{y}_2, \tau_2, \tau_2^1, \dots, \tau_2^{n-3})$ a curve in the plane starting from \mathbf{y}_1 and ending at

\mathbf{y}_2 with orientation of the tangent vector and successive derivatives with respect to s , respectively, equal to $\tau_1, \tau_1^1, \dots, \tau_1^{n-3}$ at the beginning and to $\tau_2, \tau_2^1, \dots, \tau_2^{n-3}$ at the end. This exercise is relatively easy using polynomials and transforming the boundary conditions into linear equations over the coefficients of the polynomials. However, taking into account small-time controllability is a little more difficult.

Reference [35.21] propose a flatness-based steering method built on convex combinations of canonical curves.

35.4.2 Equivalence Between Chained-Formed and Feedback-Linearizable Systems

In the previous section, we proposed methods to steer feedback-linearizable control systems or systems that can be put into chained form. We now give a necessary and sufficient condition for feedback linearizability.

Feedback Linearizability: Necessary and Sufficient Condition

In cite [35.22], *Rouchon* gives conditions to check whether a system is feedback linearizable. For two-input driftless systems a necessary and sufficient condition is the following: let us define as Δ^k , $k > 0$ the collection of distributions (i.e., the set of vector fields) iteratively defined by: $\Delta_0 = \text{span}\{f_1, f_2\}$, $\Delta_1 = \text{span}\{f_1, f_2, [f_1, f_2]\}$ and $\Delta_{i+1} = \Delta_0 + [\Delta_i, \Delta_i]$ with $[\Delta_i, \Delta_i] = \text{span}\{[f, g], f \in \Delta_i, g \in \Delta_i\}$. A system with two-dimensional input is feedback linearizable iff $\text{rank}(\Delta_i) = 2 + i$.

Example 35.1: The chained-form system. Let us consider the chained-form system defined by (35.3–35.7). The control vector fields of this system are:

$$f_1 = (1, 0, z_2, \dots, z_{n-1}) \quad f_2 = (0, 1, \dots, 0)$$

$\text{rank} \Delta_0 = 2$. If we compute $f_3 = [f_1, f_2] = (0, 0, 1, 0, \dots, 0)$, we notice that $\text{rank} \Delta_1 = 3$. By computing $f_i = [f_1, f_{i-1}]$ for i up to n we find a sequence $f_i = (0, \dots, 0, 1, 0, \dots, 0)$ where 1 is at position i . Therefore, $\text{rank} \Delta_i = 2 + i$ for i up to $n - 2$ and the chained-form system is feedback linearizable. This conclusion could have been drawn in a more straightforward way by noticing that the state can be reconstructed from (z_1, z_2) and its derivatives. (z_1, z_2) is thus the linearizing output of the chained-form system.

35.5 Robots and Trailers

Robotics systems studied in motion planning are mainly those composed of a mobile robot alone or towing one or several trailers. The input of these systems is two dimensional.

35.5.1 Differentially Driven Mobile Robots

The simplest mobile robot, namely the differentially driven mobile robot displayed in Fig. 35.1, is obviously feedback linearizable, the trajectory of the center of the wheel axis (the linearizing output) of the robot gives the orientation of the robot. It can thus be steered using a flatness-based local steering method.

35.5.2 Differentially Driven Mobile Robots Towing One Trailer

The differentially driven mobile robot towing a trailer hitched on top of the wheel axis of the robot displayed in Fig. 35.3 is feedback linearizable and the linearizing output is the center of the wheel axis of the trailer. The differentially driven mobile robot towing a trailer with a hitched kingpin (Fig. 35.4) is also feedback linearizable [35.23] but the relation between the linearizing output and configuration variables is more intricate:

$$\begin{cases} y_1 = x_1 - b \cos \theta + L(\phi) \frac{b \sin \theta + a \sin(\theta + \phi)}{\sqrt{a^2 + b^2 + 2ab \cos \phi}}, \\ y_2 = x_2 - b \sin \theta - L(\phi) \frac{a \cos(\theta + \phi) + b \cos \theta}{\sqrt{a^2 + b^2 + 2ab \cos \phi}}, \end{cases}$$

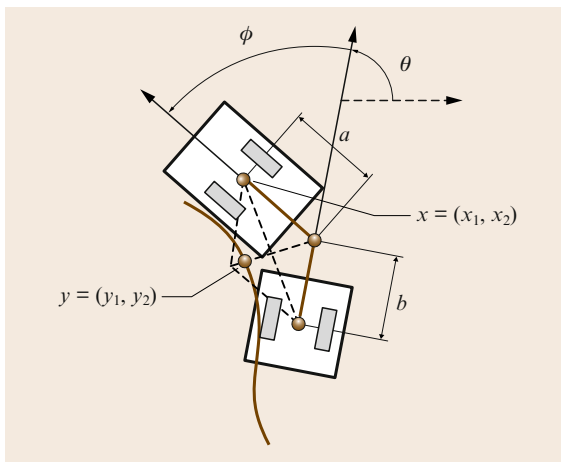


Fig. 35.4 A differentially driven mobile robot towing a trailer with a hitched kingpin is feedback linearizable

where L is defined by the following elliptic integral

$$L(\phi) = ab \int_0^\phi \frac{\cos \sigma}{\sqrt{a^2 + b^2 + 2ab \cos(\sigma)}} d\sigma,$$

which is the linearizing output of the system. These relations together with the following ones

$$\tan \tau = \frac{a \sin(\theta + \phi) + b \sin \theta}{b \cos \theta + a \cos(\theta + \phi)},$$

$$\kappa = \frac{\sin(\phi)}{\cos \phi \sqrt{a^2 + b^2 + 2ab \cos \phi} + L(\phi) \sin(\phi)}$$

make it possible to reconstruct the configuration variables from the linearizing output and its two first derivatives.

35.5.3 Car-Like Mobile Robots

A car-like mobile robot consists of a fixed rear wheel axis and two steerable front wheels, the axes of which intersect at the center of curvature (Fig. 35.5). A car-like mobile robot is kinematically equivalent to a differentially driven mobile robot towing a trailer hitched on top of the wheel axis of the robot (Fig. 35.3): the virtual front wheel corresponds to the differentially driven robot, while the body of the car corresponds to the trailer.

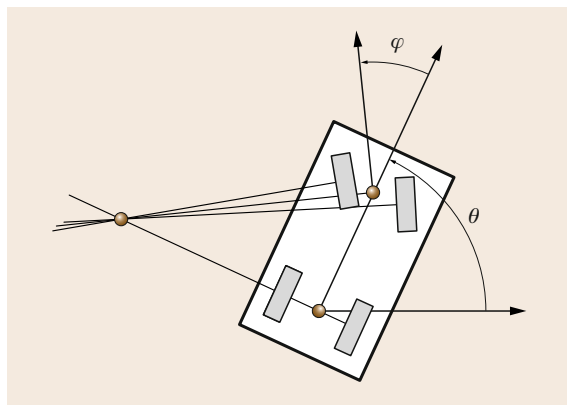


Fig. 35.5 Car-like mobile robot. The axes of the front wheels intersect at the center of curvature. The steering angle ϕ is the angle between the longitudinal axis of the car and a virtual front wheel in the middle of the two front wheels

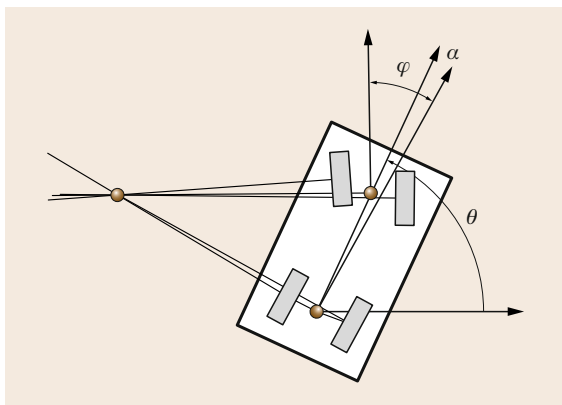


Fig. 35.6 Bi-steerable robot. The front and rear wheels are steerable and there is a relation between the rear steering angle and the front steering angle: $\alpha = f(\varphi)$

35.5.4 Bi-steerable Mobile Robots

The bi-steerable mobile robot (Fig. 35.6) is a car with front and rear steerable wheels and with a relation between the front and rear steering angles. This system has been proven to be feedback linearizable in [35.24]. As for the mobile robot towing a trailer with a hitched kingpin, the linearizing output is a moving point in the robot reference frame.

35.5.5 Differentially Driven Mobile Robots Towing Trailers

Let us consider the differentially driven mobile robot towing a trailer connected on top of the wheel axis of the robot, and let us add an arbitrary number of trailers, each one connected on top of the wheel axis of the previous one. By differentiating the curve followed by the center of the last trailer once, we get the orientation of the last trailer (this orientation coincides with the orientation of the tangent to the curve). If we know the orientation and the position of the center of the last trailer along the path, we can reconstruct the curve followed by the center of the previous trailer; repeating this reasoning, we can reconstruct the trajectory of the whole system by differentiating a sufficient number of times. The system is therefore feedback linearizable and the linearizing output is the center of the last trailer.

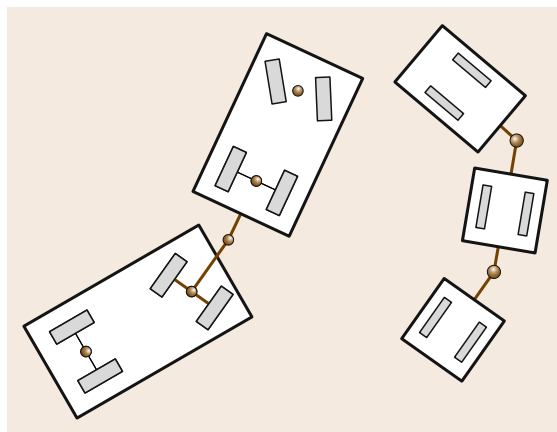


Fig. 35.7 A truck towing a towbar trailer with hitched kingpin and a differentially driven mobile robot towing two trailers with hitched kingpin: two open problems for exact path planning for nonholonomic systems

Combining the above reasoning with systems reviewed in this section, we can build hybrid feedback-linearizable trailer systems. For instance, a differentially driven mobile robot towing an arbitrary number n of trailers each one connected on top of the wheel axis of the previous trailer, except the last trailer with a hitched kingpin, is feedback linearizable. Simply consider the two last trailers as a feedback-linearizable system composed of a mobile robot with one trailer as in Fig. 35.4. The linearizing output of this system enables us to reconstruct the trajectory of the two last trailers. The center of trailer $n - 1$ is a linearizing output for the mobile robot towing the $n - 1$ last trailers.

35.5.6 Open Problems

Eventually all the systems for which we are able to plan exact motions between arbitrary pairs of configurations are included in the large class of feedback-linearizable systems. We have seen indeed that chained-form systems are also part of this class. For other systems, no exact solution has been proposed up to now, for instance, both systems displayed in Fig. 35.7 are not feedback linearizable. They do not satisfy the necessary condition of Sect. 35.4.2.

35.6 Approximate Methods

To deal with nonholonomic systems not belonging to any class of systems for which exact solutions exist, approximate numerical solutions have been developed. We review some of these methods in this section.

35.6.1 Forward Dynamic Programming

Barraquand and *Latombe* propose in [35.25] a dynamic programming approach to nonholonomic path planning. Admissible paths are generated by a sequence of constant input values, applied over a fixed interval of time δt . Starting from the initial configuration the search generates a tree: the children of a given configuration q are obtained by setting the input to a constant value and integrating the differential system over δt . The configuration space is discretized into an array of cells of equal size (i. e., hyperparallelepipeds). A child q' of a configuration q is inserted in the search tree iff the computed path from q to q' is collision free and q' does not belong to a cell containing an already generated configuration. The algorithm stops when it generates a configuration belonging to the same cell as the goal (i. e., it does not necessarily reach the goal exactly).

The algorithm has been proved to be asymptotically complete with respect to both δt and the size of the cells. As a brute force method, it remains quite time consuming in practice. Its main interest is that the search is based on Dijkstra's algorithm which allows to take into account optimality criteria such as the path length or the number of reversals. Asymptotical optimality to generate the minimum of reversals is proved for the car-like robot alone.

35.6.2 Discretization of the Input Space

Divelbiss and *Wen* propose in [35.26] a method to produce an admissible collision-free path for a nonholonomic mobile robot in the presence of obstacles. They restrict the set of input functions over the subspace spanned by a Fourier basis over the interval $[0, 1]$. An in-

put function is thus represented by a finite-dimensional vector λ . Reaching a goal configuration thus becomes a nonlinear system of equations, the unknown of which are the coordinates λ_i of λ . The authors use the Newton–Raphson method to find a solution. Obstacles are defined by inequality constraints over the configuration space. The path is discretized into N samples. The noncollision constraint expressed at these sample points yields inequality constraint over the vector λ . These inequality constraints are turned into equality constraints through the function g , defined as:

$$g(c) = \begin{cases} (1 - e^c)^2 & \text{if } c > 0 \\ 0 & \text{if } c \leq 0. \end{cases}$$

Reaching a goal configuration while avoiding obstacles thus becomes a nonlinear system of equations over vector λ again solved using the Newton–Raphson method. The method is rather efficient for short maneuvers. The main difficulty is to tune the order of the Fourier expansion. Long motions in cluttered environments require a higher order while motion in empty space can be solved with a low order. The authors do not mention the problem of numerical instability in the integration of the dynamic system.

35.6.3 Input-Based Rapidly Exploring Random Trees

RRT algorithms, described in Chap. 5, can be used without local steering method to plan paths for nonholonomic systems. New nodes can be generated from existing nodes by applying random input functions over an interval of time [35.27]. The main difficulty consists in finding a distance function that really accounts for the distance the system needs to travel to go from one configuration to another. Moreover, the goal is never exactly reached. This latter drawback can be overcome by postprocessing the path returned by **RRT** using a path deformation method, as described in [35.28].

35.7 From Motion Planning to Obstacle Avoidance

Up to now we have described motion planning techniques. Their objective is to compute a collision-free trajectory to the target configuration that complies with the vehicle constraints. They assume a perfect model

of the robot and scenario. The advantage of these techniques is that they provide complete and global solutions of the problem. Nevertheless, when the surroundings are unknown and unpredictable, these techniques fail.

A complementary way to address the motion problem is obstacle avoidance. The objective is to move a vehicle towards a target location free of collisions with the obstacles detected by the sensors during motion execution. The advantage of reactive obstacle avoidance is to compute motion by introducing the sensor information within the control loop, used to adapt the motion to any contingency incompatible with initial plans.

The main cost of considering the reality of the world during execution is locality. In this instance, if global reasoning is required, a trap situation could occur. De-

spite this limitation, obstacle avoidance techniques are mandatory to deal with mobility problems in unknown and evolving surroundings.

Notice that methods have been developed to combine both the global point of view of motion planning and the local point of view of obstacle avoidance. How to consider robot perception at the planning level? This is the so-called sensor-based motion planning. Several variants exist, such as the BUG algorithms initially introduced in [35.29]. However none of them consider the practical context of nonholonomic mobile robots.

35.8 Definition of Obstacle Avoidance

Let \mathcal{A} be the robot (a rigid object) moving in the workspace \mathcal{W} , whose configuration space is \mathcal{CS} . Let \mathbf{q} be a configuration, \mathbf{q}_t this configuration at time t , and $\mathcal{A}(\mathbf{q}_t) \in \mathcal{W}$ the space occupied by the robot in this configuration. In the vehicle there is a sensor, that in \mathbf{q}_t measures a portion of the space $\mathcal{S}(\mathbf{q}_t) \subset \mathcal{W}$ identifying a set of obstacles $\mathcal{O}(\mathbf{q}_t) \subset \mathcal{W}$.

Let \mathbf{u} be a constant control vector and $\mathbf{u}(\mathbf{q}_t)$ this control vector applied in \mathbf{q}_t during time δt . Given $\mathbf{u}(\mathbf{q}_t)$, the vehicle describes a trajectory $\mathbf{q}_{t+\delta t} = f(\mathbf{u}, \mathbf{q}_t, \delta t)$, with $\delta t \geq 0$. Let $\mathcal{Q}_{t,T}$ be the set of configurations of the trajectory followed from \mathbf{q}_t with $\delta t \in [0, T]$, a given time interval. $T > 0$ is called the *sampling period*.

Let $F : \mathcal{CS} \times \mathcal{CS} \rightarrow \mathbb{R}^+$ be a function that evaluates the progress of one configuration to another.

Obstacle Avoidance Problem

Let $\mathbf{q}_{\text{target}}$ be a target configuration. Then, in time t_i the robot \mathcal{A} is in \mathbf{q}_{t_i} , where a sensor measurement is obtained $\mathcal{S}(\mathbf{q}_{t_i})$, and thus an obstacle description $\mathcal{O}(\mathbf{q}_{t_i})$. The objective is to compute a motion control \mathbf{u}_i such that: (i) the trajectory generated is free of collisions with the obstacles $\mathcal{A}(\mathcal{Q}_{t_i,T}) \cap \mathcal{O}(\mathbf{q}_{t_i}) = \emptyset$; and (ii) it makes the vehicle progress to the target location $F(\mathbf{q}_{t_i}, \mathbf{q}_{\text{target}}) < F(\mathbf{q}_{t_i+T}, \mathbf{q}_{\text{target}})$.

The result of solving this problem at each sample time (Fig. 35.8a) is a sequence of motion controls $\{\mathbf{u}_1 \dots \mathbf{u}_n\}$ computed in execution time that avoids the obstacles gathered by the sensors, while making the vehicle progress towards the target location in each configuration $\{\mathbf{q}_{t_1} \dots \mathbf{q}_{\text{target}}\}$ (Fig. 35.8b). Notice that the motion problem is a global problem. The obstacle avoidance methods are local and iterative techniques to address this problem. The disadvantage of locality (trap situations) is counteract with the advantage of intro-

ducing the sensory information within the control cycle (to take into account the reality of the world during execution).

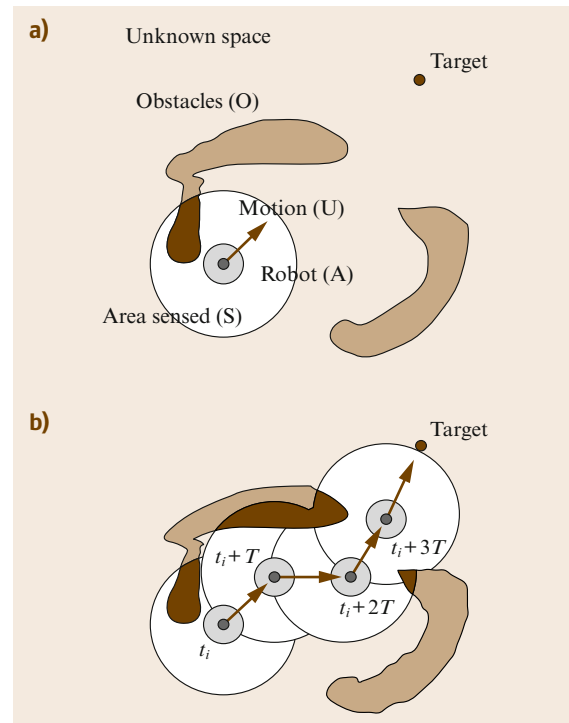


Fig. 35.8 (a) The obstacle avoidance problem consists of computing a motion control that avoid collisions with the obstacles gathered by the sensors, whilst driving the robot towards the target location. (b) The result of applying this technique at each time is a sequence of motions that drive the vehicle free of collisions to the target

There are at least three aspects that affect the development of an obstacle avoidance method: the avoidance technique, the type of robot sensor, and the type of scenario. These subjects correspond to the next three sections. First, we describe the obstacle avoidance techniques (Sect. 35.9). Second, we discuss the techniques to adapt a given obstacle avoidance method to work on a vehicle taking into account the shape, kinematics, and

dynamics (Sect. 35.10). Sensory processing is detailed in Chaps. 4 and 24. Finally, the usage of an obstacle avoidance technique on a vehicle in a given scenario is highly dependent on the scenario nature (static or dynamic, unknown or known, structured or not, or its size for example). Usually, this problem is associated with the integration of planning and obstacle avoidance (Sect. 35.11).

35.9 Obstacle Avoidance Techniques

We describe here a taxonomy of obstacle avoidance techniques and some representative methods. First there are two groups: methods that compute the motion in one step and that do it in more than one. One-step methods directly reduce the sensor information to a motion control. There are two types:

- The *heuristic* methods were the first techniques used to generate motion based on sensors. The majority of these works derived from classic planning methods and will not be described here. See [35.1, 29–32] for some representative works.
- The methods of *physical analogies* assimilate the obstacle avoidance to a known physical problem. We discuss here the potential field methods [35.33, 34]. Other works are variants adapted to uncertain models [35.35] or that use other analogies [35.36–38].

Methods with more than one step compute some intermediate information, which is processed next to obtain the motion.

- The methods of *subset of controls* compute an intermediate set of motion controls, and next choose one of them as a solution. There are two types: (i) methods that compute a subset of motion directions. We describe here the vector field histogram [35.39] and the obstacle restriction method [35.40]. Another method is presented in [35.41]. (ii) Methods that compute a subset of velocity controls. We describe here the dynamic window approach [35.42] and the velocity obstacles [35.43]. Another method based on similar principles but developed independently is the curvature velocity method [35.44].
- Finally, there are methods that compute some *high-level information* as intermediate information, which is translated next in motion. We describe the nearness diagram navigation [35.45, 46].

All the methods outlined here have advantages and disadvantages depending on the navigation context, like uncertain worlds, motion at high speeds, motion in confined or troublesome spaces, etc. Unfortunately there is no metric available to measure the performance of the methods quantitatively. However, for an experimental comparison in terms of their intrinsic problems see [35.45].

35.9.1 Potential Field Methods (PFM)

The potential field method (PFM) uses an analogy in which the robot is a particle that moves in the configuration space under the influence of a force field. While the target location exerts a force that attracts the particle \mathbf{F}_{att} , the obstacles exert repulsive forces \mathbf{F}_{rep} . At each time t_i , the motion is computed to follow the direction of the artificial force induced by the sum of both potentials $\mathbf{F}_{\text{tot}}(\mathbf{q}_{t_i}) = \mathbf{F}_{\text{att}}(\mathbf{q}_{t_i}) + \mathbf{F}_{\text{rep}}(\mathbf{q}_{t_i})$ (the most promising motion direction), Fig. 35.9.

Example 35.2:

$$\mathbf{F}_{\text{att}}(\mathbf{q}_{t_i}) = K_{\text{att}} \mathbf{n}_{\mathbf{q}_{\text{target}}} \quad (35.9)$$

$$\mathbf{F}_{\text{rep}}(\mathbf{q}_{t_i}) = \begin{cases} K_{\text{rep}} \sum_j \left(\frac{1}{d(\mathbf{q}_{t_i}, \mathbf{p}_j)} - \frac{1}{d_0} \right) \mathbf{n}_{\mathbf{p}_j}, & \text{if } d(\mathbf{q}_{t_i}, \mathbf{p}_j) < d_0 \\ 0, & \text{otherwise} \end{cases} \quad (35.10)$$

where K_{att} and K_{rep} are the constants of the forces, d_0 is the influence distance of the obstacles \mathbf{p}_j , \mathbf{q}_{t_i} is the current vehicle configuration, and $\mathbf{n}_{\mathbf{q}_{\text{target}}}$ and $\mathbf{n}_{\mathbf{p}_j}$ are the unitary vectors that point from \mathbf{q}_{t_i} to the target and each obstacle \mathbf{p}_j , respectively. From $\mathbf{F}_{\text{tot}}(\mathbf{q}_{t_i})$, the control \mathbf{u}_i can be obtained with a position or force control [35.47].

This is the classic version, where the potentials only depend on the current vehicle configuration. Comple-

mentarily the generalized potentials depend also on the instantaneous robot velocity and vehicle accelerations.

Example 35.3:

$$F_{\text{rep}}(q_{t_i}) = \begin{cases} K_{\text{rep}} \sum_j \left(\frac{a \dot{q}_{t_i}}{(2ad(q_{t_i}, p_j) - \dot{q}_{t_i}^2)} \right) n_{p_j} \cdot n_{\dot{q}_{t_i}}, & \text{if } \dot{q}_{t_i} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (35.11)$$

where \dot{q}_{t_i} is the current robot velocity, $n_{\dot{q}_{t_i}}$ the unitary vector pointing in the direction of the robot velocity, and a is the maximum vehicle acceleration. This expression arises when the repulsive potential is defined as the inverse of the difference between the estimated time until a collision takes place and the time needed to stop

the robot until full backward acceleration. Notice how the repulsive force only affects in the direction of motion of the vehicle as opposed to the classic potential. For a comparison between the classic and generalized versions, and the relation with the way to compute the motion controls next, see [35.47]. This method is widely used due to its easy understanding and clear mathematic formalism.

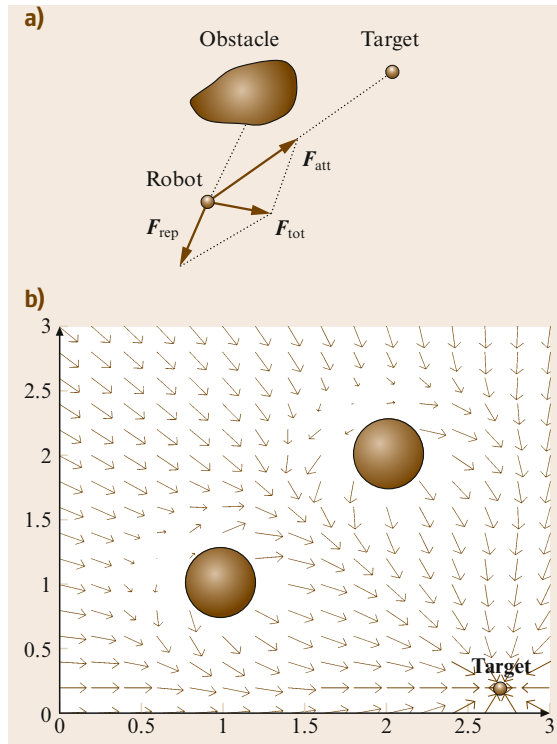


Fig. 35.9 (a) Computation of the motion direction with a potential field method. The target attracts the particle F_{att} while the obstacle exerts a repulsive force F_{rep} . The resulting force F_{tot} is the most promising motion direction. (b) Motion directions computed in each point of the space with the classic method

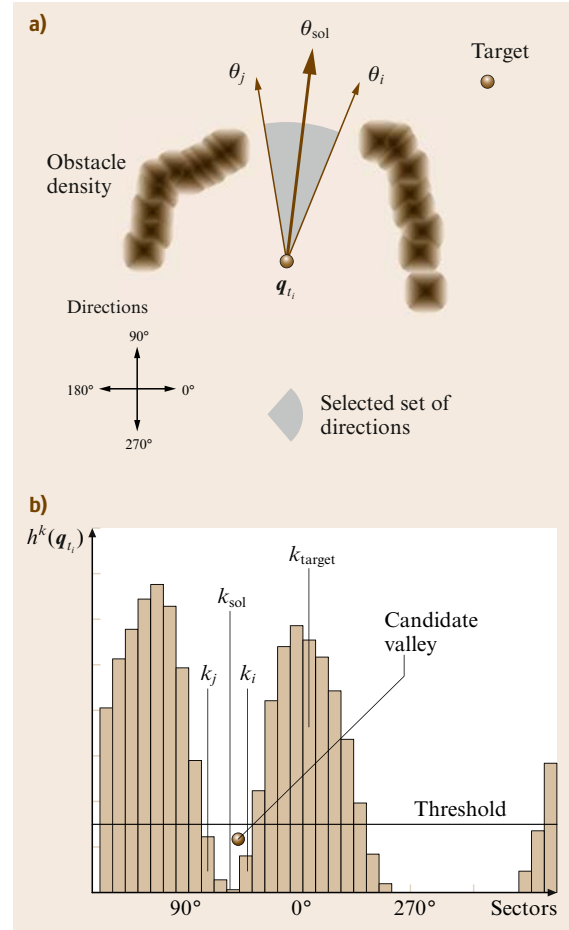


Fig. 35.10a,b Computation of the motion direction θ_{sol} with the VFH. (a) Robot and obstacle occupancy distribution. (b) The candidate valley is the set of adjacent components with lower value than the threshold. The navigation case is case 3 since the sector of the target k_{target} is not in the valley and the number of sectors is lower than a fixed quantity m ($m = 8$, i. e., 45°). Thus the solution is $k_{\text{sol}} = \frac{k_i + k_j}{2}$, whose bisector is θ_{sol} in (a). The bisectors of k_i and k_j are θ_i and θ_j , respectively

35.9.2 Vector Field Histogram (VFH)

The **VFH** solves the problem in two steps by computing a set of candidate motion directions and then selecting one of them.

Candidate Set of Directions

Firstly the space is divided into sectors from the robot location. The method uses a polar histogram H constructed around the robot, where each component represents the obstacle polar density in the corresponding sector. The function that maps the obstacle distribution in sector k on the corresponding component of the histogram $h^k(q_{t_i})$ is:

$$h^k(q_{t_i}) = \int_{\Omega_k} P(p)^n \left(1 - \frac{d(q_{t_i}, p)}{d_{\max}} \right)^r dp \quad (35.12)$$

The domain of integration is $\Omega_k = \{p \in \mathcal{W} \setminus p \in k \wedge d(q_{t_i}, p) < d_0\}$. The density $h^k(q_{t_i})$ is proportional to the probability $P(r)$ that a point was occupied by an obstacle, and to a factor that increases as the distance to the point decreases (both functions powered by some integers $n, r > 0$).

Typically the resulting histogram has peaks (directions with a high density of obstacles) and valleys (directions with a low density). The set of candidate directions is the set of adjacent components with lower density than a given threshold, and closest to the component that contains the target direction. This set of components (sectors) is called the selected valley, and represents a set of candidate directions (Fig. 35.10).

Motion Computation

The objective of the next step is to choose a direction of this set. The strategy is to apply three heuristics that depend on the component that contains the target or on the size of the selected valley. The cases are checked in sequence:

1. Case 1: goal sector in the selected valley. Solution: $k_{\text{sol}} = k_{\text{target}}$, where k_{target} is the sector that contains the goal location.
2. Case 2: goal sector not in the selected valley and a number of sectors of the valley greater than m . Solution: $k_{\text{sol}} = k_i \pm \frac{m}{2}$, where m is a fixed number of sectors and k_i the sector of the valley closer to the k_{target} .
3. Case 3: goal sector not in the selected valley and number of sectors of the valley lower or equal to m . Solution: $k_{\text{sol}} = \frac{k_i + k_j}{2}$, where k_i and k_j are the extreme sectors of the valley.

The result is a sector k_{sol} whose bisector is the direction solution θ_{sol} . The velocity v_{sol} is inversely proportional to the distance to the closest obstacle. The control is $u_i = (v_{\text{sol}}, \theta_{\text{sol}})$.

The **VFH** is a method formulated to work with probability obstacle distributions and thus is well adapted to work with uncertain sensors such as ultrasonic sonars.

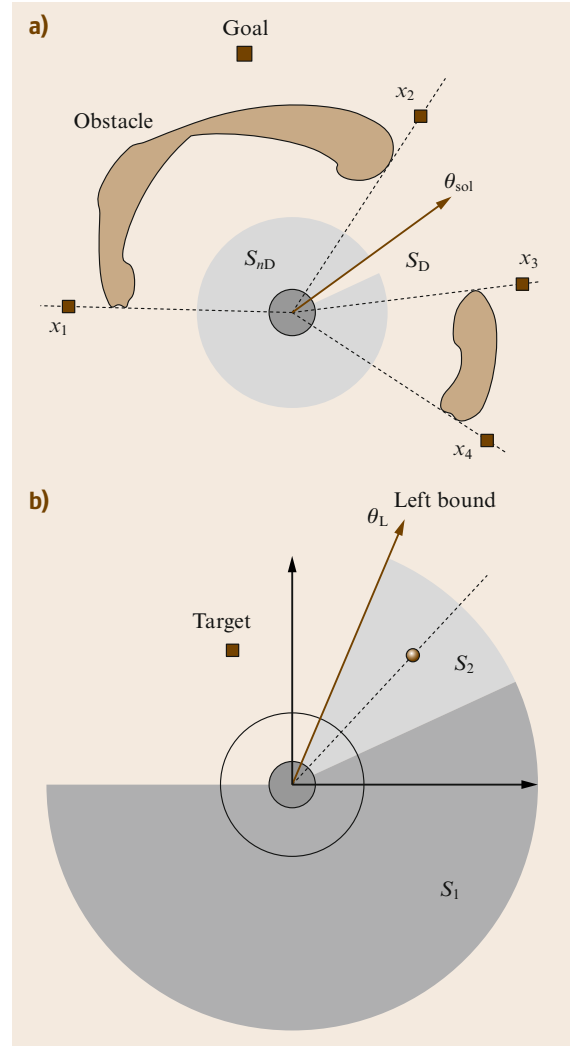


Fig. 35.11 (a) Distribution of the subgoals x_i . The selected instantaneous target location is x_2 . The set of candidate directions is S_{nD} and the solution is θ_{sol} (the second case). (b) The two set of undesirable directions S_1 and S_2 for a given obstacle

35.9.3 The Obstacle Restriction Method (ORM)

The **ORM** solves the problem in three steps, where the result of the first two steps is a set of candidate motion directions. The first step is to compute an instantaneous subgoal if necessary. The second step then associates a motion constraint to each obstacle and joins them next to compute the set of desirable directions. The last step is a strategy to compute the motion given this set.

Instantaneous Target Selection

This step computes a subgoal when is better to direct the motion towards a given zone of the space (that ameliorates the situation to reach the goal latter), rather than directly towards the goal itself. The subgoals are located in between obstacles or at the edge of an obstacle (Fig. 35.11a). Next, the process checks by using a local algorithm whether the goal can be reached from the robot location. If it cannot, then the closest reachable subgoal to the goal is selected. To check whether a point can be reached, there is a local algorithm that computes the existence of a local path that joins two locations [35.45]:

Let \mathbf{x}_a and \mathbf{x}_b be two locations of the space, R the robot radius, and L a list of obstacle points, where \mathbf{p}_i is an obstacle of the list. Let \mathcal{A} and \mathcal{B} be the two semiplanes divided by the line that joins \mathbf{x}_a and \mathbf{x}_b . Then, if for all the points of L , $d(\mathbf{p}_j, \mathbf{p}_k) > 2R$ (with $\mathbf{p}_j \in \mathcal{A}$ and $\mathbf{p}_k \in \mathcal{B}$), then there is a collision-free path that joins both locations. If this condition is not satisfied, then there is no local path (although a global one could exist). The interesting result is when the result is positive, since the guarantee that one point can be reached from the other exists.

The result of the process is the goal or an instantaneous subgoal (from now on this location is called the target location). Notice that this process is general and can be used as a preprocessing step by other methods to instantaneously validate the goal location or to compute an instantaneous subgoal to drive the vehicle.

Candidate Set of Directions

For each obstacle i a set of not desirable directions for motion S_{nD}^i is computed (the motion constraint). This set is the union of two subsets: S_1^i and S_2^i . S_1^i represents the side of the obstacle not suitable to achieve avoidance and S_2^i is an exclusion area around the obstacle (Fig. 35.11b). The motion constraint for the obstacle is the union of these two sets $S_{nD}^i = S_1^i \cup S_2^i$. The set of

desired directions of motion is the complementary set $S_D = \{[-\pi, \pi] \setminus dS_{nD}\}$, where $S_{nD} = \cup_i S_{nD}^i$.

Motion Computation

The final step is to select a direction of motion. There are three cases, depending on the set of desirable directions S_D and on the target direction θ_{target} . The cases are checked in sequence:

1. Case 1: $S_D \neq \emptyset$ and $\theta_{\text{target}} \in S_D$. Solution: $\theta_{\text{sol}} = \theta_{\text{target}}$.
2. Case 2: $S_D \neq \emptyset$ and $\theta_{\text{target}} \notin S_D$. Solution: $\theta_{\text{sol}} = \theta_{\text{lim}}$, where θ_{lim} is the direction of S_D closest to θ_{target} .
3. Case 3: $S_D = \emptyset$. Solution: $\theta_{\text{sol}} = \frac{\phi_{\text{lim}}^l + \phi_{\text{lim}}^r}{2}$, where ϕ_{lim}^l and ϕ_{lim}^r are the directions of S_{D_r} and S_{D_l} closer to θ_{target} (S_{D_r} and S_{D_l} are the set of desirable directions of obstacles on the left-hand side and right-hand side of the target, respectively).

The result is a solution θ_{sol} for the direction of motion. The velocity v_{sol} is inversely proportional to the distance to the closest obstacle. The control is $\mathbf{u}_i = (v_{\text{sol}}, \theta_{\text{sol}})$.

This is a geometric-based method based on cases. The advantage is that it has been demonstrated to address effective motion in confined spaces.

35.9.4 Dynamic Window Approach (DWA)

The **DWA** is a method that solves the problem in two steps, computing as intermediate information a subset of the control space \mathcal{U} . For simplicity, we consider

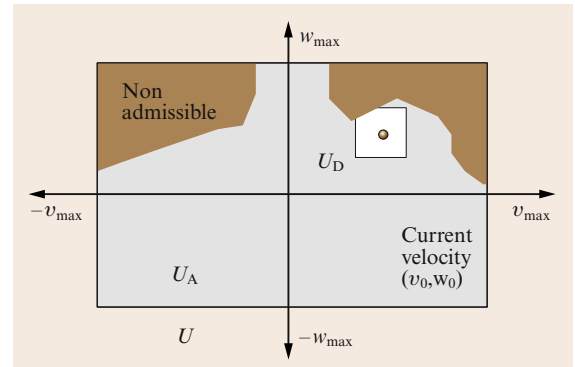


Fig. 35.12 Subset of controls $\mathcal{U}_R = \mathcal{U} \cap \mathcal{U}_A \cap \mathcal{U}_D$, where \mathcal{U} contains the controls within the maximum velocities, \mathcal{U}_A the admissible controls, and \mathcal{U}_D the controls reachable by a short period of time

a motion control as translational and rotational velocity (v, w) . \mathcal{U} is defined by:

$$\mathcal{U} = \{(v, w) \in \mathbb{R}^2 \mid v \in [-v_{\max}, v_{\max}] \wedge w \in [-w_{\max}, w_{\max}]\}. \quad (35.13)$$

Set of Candidate Controls

The candidate set of controls \mathcal{U}_R contains the controls that (i) are within the maximum velocities of the vehicle \mathcal{U} ; (ii) generate safe trajectories \mathcal{U}_A ; and (iii) can be reached within a short period of time given the vehicle accelerations \mathcal{U}_D . The set \mathcal{U}_A contains the admissible controls. These controls can be canceled before collision by applying the maximum deceleration (a_v, a_w) :

$$\mathcal{U}_A = \{(v, w) \in \mathcal{U} \mid v \leq \sqrt{2d_{\text{obs}}a_v} \wedge w \leq \sqrt{2\theta_{\text{obs}}a_w}\}, \quad (35.14)$$

where d_{obs} and θ_{obs} are the distance to the obstacle and the orientation of the tangent to the trajectory over the obstacle, respectively. The set \mathcal{U}_D contains the reachable controls in a short period:

$$\mathcal{U}_D = \{(v, w) \in \mathcal{U} \mid v \in [v_0 - a_v T, v_0 + a_v T] \wedge w \in [w_0 - a_w T, w_0 + a_w T]\}, \quad (35.15)$$

where $\dot{\mathbf{q}}_{t_i} = (v_0, w_0)$ is the current velocity.

The resulting subset of controls is (Fig. 35.12):

$$\mathcal{U}_R = \mathcal{U} \cap \mathcal{U}_A \cap \mathcal{U}_D. \quad (35.16)$$

Motion Computation

The next step is to select one control $\mathbf{u}_i \in \mathcal{U}_R$. The problem is set out as the maximization of an objective function:

$$G(\mathbf{u}) = \alpha_1 \cdot \text{Goal}(\mathbf{u}) + \alpha_2 \cdot \text{Clearance}(\mathbf{u}) + \alpha_3 \cdot \text{Velocity}(\mathbf{u}). \quad (35.17)$$

This function is a compromise among $\text{Goal}(\mathbf{u})$, which favors velocities that offer progress to the goal, $\text{Clearance}(\mathbf{u})$, which favors velocities far from the obstacles, and $\text{Velocity}(\mathbf{u})$ that favors high speeds. The solution is the control \mathbf{u}_i that maximizes this function.

The **DWA** solves the problem in the control space using information of the vehicle dynamics, thus the method is well adapted to work on vehicles with slow dynamic capabilities or that work at high speeds.

35.9.5 Velocity Obstacles (VO)

The velocity obstacles (**VO**) method solves the problem in two steps, by computing as intermediate information

a subset of the \mathcal{U} . The framework is the same as for the **DWA** method. The difference is that the computation of the set of safe trajectories \mathcal{U}_A takes into account the velocity of the obstacles, which is described next.

Let \mathbf{v}_i be the velocity of obstacle i (that enlarged with the vehicle radius occupies an area B_i) and \mathbf{u} a given vehicle control. The set of colliding relative velocities is called the *collision cone*:

$$CC_i = \{\mathbf{u}_i \mid \lambda_i \cap B_i \neq \emptyset\}, \quad (35.18)$$

where λ_i is the direction of the unitary vector $\mathbf{u}_i = \mathbf{u}_i - \mathbf{v}_i$. The *velocity obstacle* is this set in a common absolute system of reference:

$$VO_i = CC_i \oplus \mathbf{v}_i, \quad (35.19)$$

where the \oplus is the Minkowski vector sum. The set of unsafe trajectories is the union of the velocity obstacles for each moving obstacle $\tilde{\mathcal{U}}_A = \cup_i VO_i$ (Fig. 35.13).

The advantage of this method is that it takes into account the velocity of the obstacles and thus is well suited to dynamic scenarios.

35.9.6 Nearness Diagram Navigation (ND)

This method is more a methodology to design obstacle avoidance methods rather than a method in itself. The **ND** is an obstacle avoidance method obtained with a geometric implementation following this methodology. The idea behind this approach is to employ a divide-

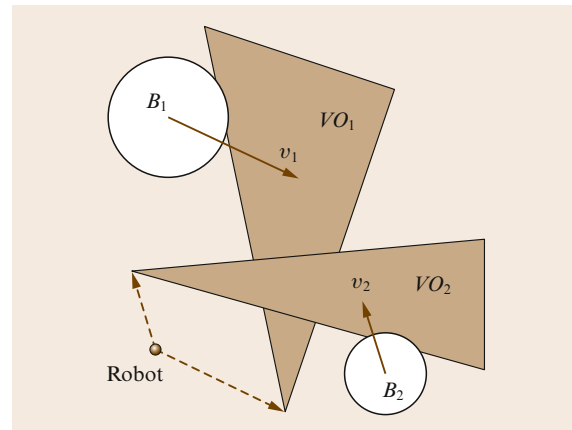


Fig. 35.13 The subset of controls that are not safe $\tilde{\mathcal{U}}_A = VO_1 \cup VO_2$. A control vector out of this set generates a collision free-motion with the moving obstacles

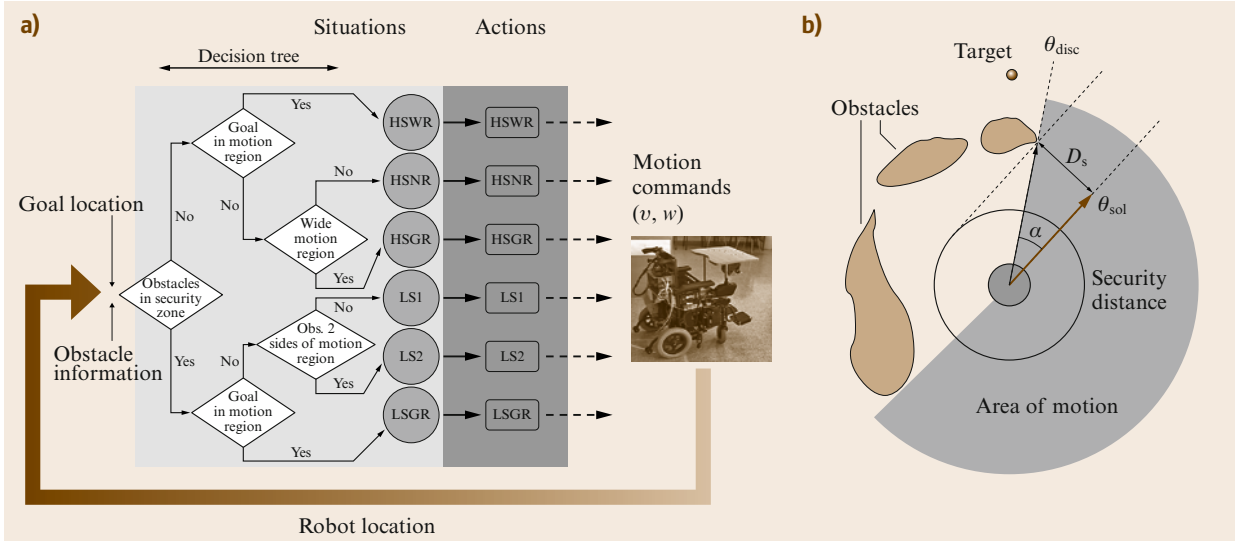


Fig. 35.14 (a) Design diagram of the method. Given the obstacle information and the target location, one situation is identified given some criteria. Next the associated action is executed, computing the motion. (b) Example of computation of the solution of the ND (geometric implementation). The first step is to identify the situation. There are not obstacles closer than the security distance D_s . Next the q_{target} is not within the motion area. Third, the motion area is wide. With these three criteria we identify the current situation, HSWR. In this situation, the associated action computes the control as $u_i = (v_{sol}, \theta_{sol})$, where v_{sol} is the maximum velocity and θ_{sol} is computed as a deviation α from the limit direction θ_{disc} of the motion area closer to the target direction

and-conquer strategy based on situations to simplify the obstacle avoidance problem following the situated-activity paradigm (see [35.48] for a review). First, there is a set of situations that represent all the cases among robot locations, obstacles, and target locations. Also, a motion law is associated with each case. During the execution phase, at time t_i , one situation is identified and the corresponding law is used to compute the motion.

Situations

The situations are represented in a binary decision tree. The selection of a situation depends on the obstacles $\mathcal{O}(q_i)$, on the robot location q_i , and the target q_{target} . The criteria are based on high-level entities like a security distance around the robot bounds and a motion area (that identifies suitable areas of motion), for example, one criterion is whether there are obstacles within the security zone. Another is whether the motion area is large or narrow. The result is only one situation since, by definition and representation (binary decision tree), the set of situations is complete and exclusive (Fig. 35.14).

Actions

Associated to any situation there is an action that computes the motion to adapt the behavior to the case represented by each situation. At a high level the actions describe the behavior desired in each situation, for example, one situation is when there are no obstacles within a security area and the goal is within the motion area (HSGR). The solution is to *move towards the goal*. Another situation is when there are no obstacles within a security area, and the goal is not within the motion area but is wide (HSWR). The solution is to *move towards the limit of the motion area but clearing the security area of obstacles*.

The interesting aspect of this method is that it employs a *divide-and-conquer* strategy to solve the navigation and thereby simplifies the problem. The first advantage is that the methodology is described at a symbolic level and thus there are many ways to implement the method. Second, a geometric implementation of this methodology (the ND method) has been demonstrated to address difficult navigation cases that translate to achieving safe navigation in dense, complex, and difficult scenarios.

35.10 Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance

There are three aspects of the vehicle that have to be taken into account during the obstacle avoidance process: shape, kinematics, and dynamics. The shape and the kinematics together form a geometric problem that involves the representation of the vehicle configurations in collision given the admissible trajectories $\mathcal{Q}_{t,\infty}$. The

dynamics involves the accelerations and temporal considerations and has two aspects: (i) choosing a control reachable in a short period of time T given the current velocity \dot{q}_{t_i} and the maximum accelerations; (ii) taking into account the braking distance, so that after a control execution, the vehicle can always stop before collision by applying the maximum deceleration (improving safety).

The problem of the shape, kinematics, and dynamics in obstacle avoidance has been taken into account from three different points of view: (i) designing a way to incorporate the constraints within the method (Sect. 35.9.4), (ii) developing techniques that abstract the vehicle aspects from the application of the method [35.49, 50], and (iii) by techniques that break down the problem into subproblems and incorporate the aspects in sequence [35.51–53] after the method usage.

35.10.1 Techniques that Abstract the Vehicle Aspects

These techniques are based on constructing an abstraction layer between the aspects of the vehicle and the obstacle avoidance method in such a way that, when the method is applied, its solutions already take into account these aspects [35.49, 50]. We consider here vehicles whose elementary paths obtained under the execution of constant controls can be approximated by circular arcs (for example a differential-drive robot, a synchrodrive or a tricycle). To simplify the analysis, a control is a translational and rotational velocity $u = (v, w)$. The set of reachable controls \mathcal{U}_A given the current velocity $\dot{q}_{t_i} = (v_0, w_0)$ and the maximum accelerations (a_v, a_w) is obtained by (35.15).

Abstraction Construction

For these vehicles, the configuration space \mathcal{CS} is three dimensional. The idea is to construct, centered at the robot at each time t_i , the manifold of the configuration space $ARM(q_{t_i}) \equiv ARM$ defined by elementary circular paths. The function that defines the manifold is:

$$\theta = f(x, y) = \begin{cases} \arctan 2\left(x, \frac{x^2 - y^2}{2y}\right) & \text{if } y \geq 0, \\ -\arctan 2\left(x, -\frac{x^2 - y^2}{2y}\right) & \text{otherwise.} \end{cases} \quad (35.20)$$

It is easy to see that the function f is differentiable in $\mathbb{R}^2 \setminus (0, 0)$. Thus $(x, y, f(x, y))$ de-

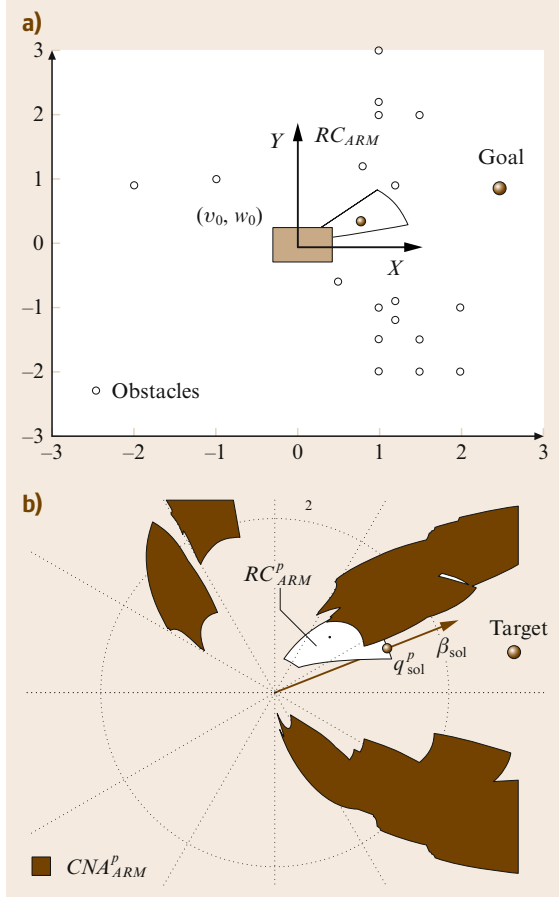


Fig. 35.15 (a) Robot location, obstacle information, and goal location. (b) The abstraction layer in ARM^p . In this representation the obstacles are the no admissible region CNA_{ARM}^p in ARM^p and motion is omnidirectional (applicability conditions of many obstacle avoidance methods). The method is applied to obtain the most promising direction β_{sol} , used to obtain the configuration solution q_{sol}^p in the set of reachable configurations RC_{ARM}^p . Finally, the solution is the control u_{sol} that reaches this configuration at time T . This control complies with the kinematics and dynamics and takes into account the exact vehicle shape

defines a two-dimensional manifold in $\mathbb{R}^2 \times \mathcal{S}^1$ when $(x, y) \in \mathbb{R}^2 \setminus (0, 0)$. This manifold ARM contains all the configurations that can be reached at each step of the obstacle avoidance.

Next, in the ARM one computes the exact region of the configurations in collision CO_{ARM} given any shape of the robot (i. e., obstacle representation in the manifold). Given an obstacle point (x_p, y_p) and a point of the robot bounds (x_r, y_r) , a point (x_s, y_s) of the CO_{ARM} boundary is obtained by

$$\begin{aligned} x_s &= (x_f + x_i) a, \\ y_s &= (y_f - y_i) a, \end{aligned} \quad (35.21)$$

with

$$a = \frac{[(y_f^2 - y_i^2) + (x_f^2 - x_i^2)][(y_f - y_i)^2 + (x_f - x_i)^2]}{(y_f - y_i)^4 + 2(x_f^2 + x_i^2)(y_f - y_i)^2 + (x_f^2 - x_i^2)^2}.$$

This result is used to map the robot bounds for all obstacles in the manifold, computing the exact shape of CO_{ARM} . Next, one computes the nonadmissible configurations CNA_{ARM} in the manifold ARM , which correspond to configurations where, once reached at a given velocity in time T , the vehicle cannot be stopped by applying the maximum deceleration without collision (i. e., there is not enough braking distance). These regions CNA_{ARM} are the CO_{ARM} enlarged by a magnitude that depends on the maximum vehicle accelerations. The set of configurations reachable RC_{ARM} by controls reachable in a short period of time \mathcal{U}_A is also computed in the ARM . Finally, a change of coordinates is applied to ARM , leading to ARM^p ; its effect is that elementary circular paths become straight segments in the manifold. As a consequence, the problem is now to move an omnidirectional point in a bidimensional space free of any constraint.

Method Application

The final step is to apply the avoidance method on ARM^p to avoid the CNA_{ARM}^p regions. The method solution β_{sol} is the most promising direction in ARM^p . This direction is used to select an admissible location $q_{sol}^p \notin CNA_{ARM}^p$ in the set of dynamic reachable configurations RC_{ARM}^p . Finally, the control u_{sol} that leads to this location at time T is selected. By construction, this control is kinematic and dynamic admissible, avoids collisions with the exact vehicle shape and takes into account the braking distance (Fig. 35.15).

Notice how with these techniques, the three-dimensional obstacle avoidance problem is converted into the simple problem of moving a point in a two-dimensional space without kinematic and dynamic restrictions (the applicability conditions for many existing methods). Thus, many existing methods can be applied in this abstraction, and as a consequence, the solutions take into account the vehicle constraints.

35.10.2 Techniques of Decomposition in Subproblems

These techniques address the obstacle avoidance problem, taking into account the vehicle constraints by breaking down this problem into subproblems: (i) obstacle avoidance, (ii) kinematics and dynamics, and (iii) shape. Each subproblem is dealt with in sequence (Fig. 35.16).

Obstacle Avoidance

First the obstacle avoidance method is used by assuming a circular and omnidirectional vehicle. The solution is the most promising motion direction and the velocity $u_1 = (v_1, \theta_1)$ to direct the vehicle towards the target.

Kinematics and Dynamics

Second, this control is converted into a control that complies with the kinematics and dynamics, which tends to align the vehicle with the instantaneous motion direction θ_1 of u_1 , for example, [35.52, 53] modify the output of the obstacle avoidance method by a feedback action that aligns the vehicle with the direction solution in a least-squares fashion; or [35.51] use a dynamic controller of the vehicle. This controller models the behavior of the vehicle as if it were pulled by a virtual force, and computes the motion that would result after applying this force during a short period of time. The motion generated is the new control. In order to use the controller, the previous control u_1 is converted into an instantaneous force

Steps	Step 1	Step 2	Step 3
Robot aspects	Circular omnidirectional	Circular with kinematics and dynamics	Rectangular with kinematics and dynamics
Subproblem (solution)	Obstacle avoidance	Motion controller	Shape corrector

Fig. 35.16 Decomposition of the obstacle avoidance problem into subproblems that are addressed in steps, incorporating the vehicle aspects successively

$\mathbf{F} = (\theta_1, F_{\max} \frac{v_1}{v_{\max}})$ input of the controller, that computes a control \mathbf{u}_2 that complies with the kinematic and dynamics.

Shape

The final step is to ensure that the control \mathbf{u}_2 obtained in the previous step avoids collisions with the exact shape

of the vehicle. To do this, a shape corrector is used to check collisions by dynamic simulation of the control. If there is a collision, the set of reachable controls (35.15) is explored until a collision-free control is found. The result of this process is a motion control \mathbf{u}_i that guarantees obstacle avoidance and complies with the kinematics and dynamics of the vehicle.

35.11 Integration Planning – Reaction

In this section we show how the obstacle avoidance methods are integrated into real systems. On the one hand, obstacle avoidance methods are local techniques to address the motion problem. Thus, they are doomed to fall into local minima that translates in trap situations or cyclic motions. This reveals the necessity of more-global reasoning. On the other hand, motion planning techniques compute a geometric path free of collisions that guarantees global convergence. However, when the scenarios are unknown and evolve, these techniques fail, since the precomputed paths will almost surely collide with obstacles. It seems clear that one key aspect to build a motion system is to combine the best of both worlds: the global knowledge given by motion planning and the reactivity of obstacle avoidance methods.

The most extended ways to specify the interaction between the deliberation and reaction are: (i) to precompute a path to the target, which is then deformed during execution as a function of changes in the scenario obtained from sensor information (path deformation systems), for example, [35.54–56]. (ii) To use a planner at a high frequency with a tactical role, leaving the degree of execution to the reactor [35.57–62].

35.11.1 Path Deformation Systems

The *elastic bands* method initially assumes the existence of a geometric path to the target location (computed by a planner). The path is assimilated with a band, subjected to two types of forces: an inner contraction force and an external force. The inner force simulates the tension of a strip and maintains the stress. The external force is exerted by the obstacles and moves the band away from them. During execution, new obstacles produce forces that move the band away from them, guarantying their avoidance. These methods are described in Chap. 5 of this book.

An extension of the method of path deformation has been proposed in [35.56] for nonholonomic sys-

tems. Even though the objective is the same as for mobile robots without kinematic constraints (avoiding obstacles while following a trajectory), the concepts are completely different. The trajectory Γ of a nonholonomic system is completely defined by an initial configuration $\Gamma(0)$ and the value of the input function $\mathbf{u} \in C^1(I, \mathbb{R}^m)$, where I is an interval. The trajectory deformation method for a nonholonomic system is thus based on the perturbation of the input function of the current trajectory in order to achieve three objectives:

1. keeping the nonholonomic constraints satisfied
2. getting away from obstacles detected online by on-board sensors
3. keeping unchanged the initial and last configurations of the trajectory after deformation

Perturbing the input function of Γ , by a vector-valued input perturbation $\mathbf{v} \in C^1(I, \mathbb{R}^m)$ yields a trajectory deformation $\eta \in C^1(I, \mathbb{R}^n)$:

$$\mathbf{u} \leftarrow \mathbf{u} + \tau \mathbf{v} \Rightarrow \Gamma \leftarrow \Gamma + \tau \eta, \quad (35.22)$$

where τ is a positive asymptotically small real number. As an approximation of order one, the relation between \mathbf{u} and η is given by the linearized system, which we do not express here.

Obstacle Potential Field

Obstacles are taken into account by defining a potential field over the configuration space, which increases when the robot gets closer to obstacles. This potential field is lifted into the space of trajectories by integration along the trajectory of the configuration potential value.

Discretization of the Input Space

The space $C^1(I, \mathbb{R}^m)$ of input perturbations is an infinite-dimensional vector space. The choice of an input perturbation is restricted to a finite-dimensional sub-

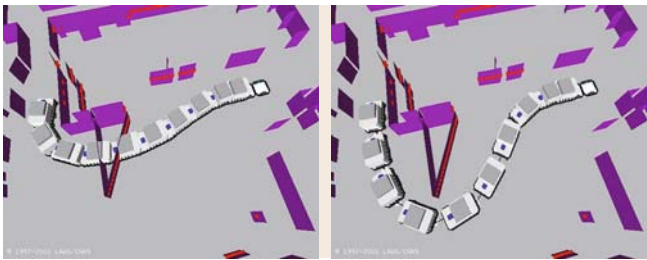


Fig. 35.17 A differentially driven mobile robot towing a trailer, applying the trajectory deformation algorithm to avoid obstacles detected online

space spanned by p arbitrary test functions, e_1, \dots, e_p , where p is a positive integer. An input perturbation: $u = \sum_{i=1}^p \lambda_i e_i$ is thus defined by a vector $\lambda \in \mathbb{R}^p$. The variation of the trajectory potential is linear with respect to λ .

Boundary Conditions

The boundary conditions consist of applying a trajectory perturbation equal to zero at both ends of the interval I that is linear with respect to λ . It is therefore easy to find a vector λ that makes the potential decrease and that satisfies the boundary conditions.

Nonholonomic Constraint Deviation

The approximation of order one in the relation between the input perturbation and trajectory deformation induces a side effect: after a few iterations, the nonholonomic constraints are not satisfied anymore. This side effect is corrected by considering an augmented system with n control vector fields f_1, \dots, f_n spanning \mathbb{R}^n for each configuration and by keeping the input components

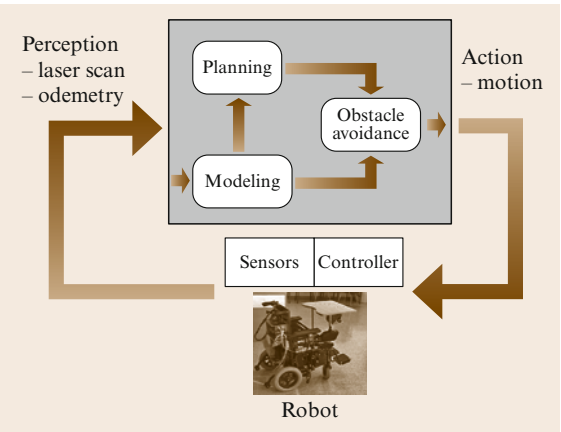


Fig. 35.18 Overview of the motion system

u_{m+1} to u_n along the additional vector fields as close as possible to zero.

Figure 35.17 shows an example of the trajectory deformation algorithm applied to a nonholonomic system.

35.11.2 Tactical Planning Systems

Tactical planning systems recompute a path to the target location at a high frequency, and use the main course to advise the obstacle avoidance module. The design of these motion systems involves the synthesis of (at least) three functionalities: the construction of a model, deliberative planning, and obstacle avoidance. The mod-

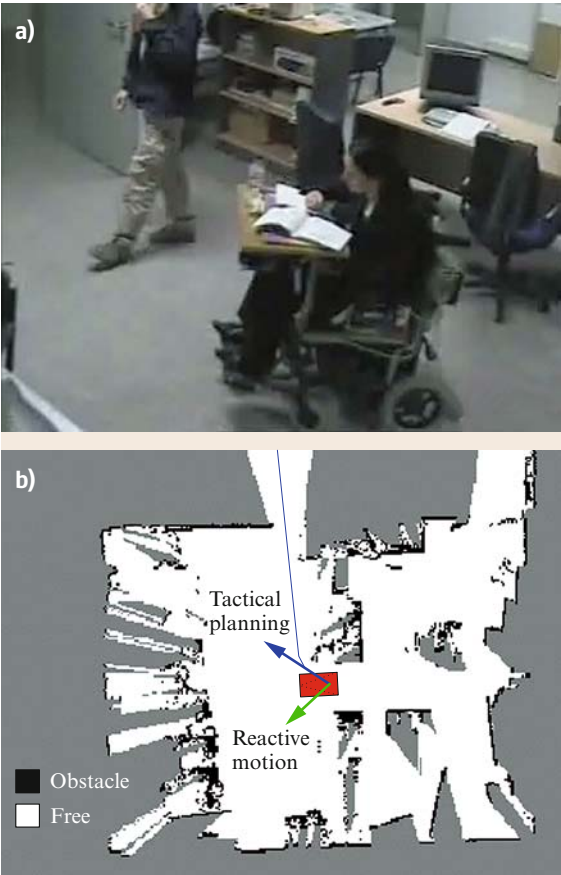


Fig. 35.19 (a) Snapshot of an experiment carried out with the motion system described working on a wheelchair robot equipped with a planar range laser sensor. (b) Information of the motion system at a given time: current accumulated map of the scenario, the path computed with the planner and tactical direction of motion, and the solution of the reactive obstacle avoidance method

eler constructs a representation base for deliberation and memory for the reactive behavior. The planner generates global plans that are used as tactics to guide the obstacle avoidance module, which generates the local motion. Below we give a perspective of the three functionalities and three possible tools to implement them [35.62].

Model Builder Module

The construction of a model of the environment (to increase the spatial domain of the planning and used as a local memory for obstacle avoidance): one possibility is to use a binary occupancy grid that is updated whenever a new sensory measurement is available, and to employ a scan matching technique [35.63, 64] to improve the vehicle odometry before integrating any new measure in the grid.

Planner Module

Extraction of the connectivity of the free space (used to avoid the cyclical motions and trap situations): a good choice is a dynamic navigation function such as D^* [35.65, 66]. The idea behind the planner is to focus the search locally in the areas where the changes in the scenario structure have occurred and affect the computation of the path. The planner avoids the local minima and is computationally very efficient for real-time implementations.

Obstacle Avoidance Module

Computation of the collision-free motion: any of the methods described in this chapter could be used. One possibility is the ND method (Sect. 35.9.6) since it has been demonstrated to be very efficient and robust in environments with little space to manoeuvre.

Globally the system works as follows (Fig. 35.18): given a laser scan and the odometry of the vehicle, the model builder incorporates this information into the existing model. Next, the information about changes in obstacle and free space in the model is used by the planner module to compute the course to follow to reach the goal. Finally, the avoidance module uses the information about the obstacles contained in the grid and information from this tactical planner to generate the motion (to drive the vehicle free of collisions towards the goal). The motion is executed by the vehicle controller and the process restarts with a new sensorial measurement. It is important to stress that the three modules should work synchronously within the perception-action cycle for consistency reasons. The advantage of these systems is that the synergy of the modules allows the avoidance of trap situations and cyclic motions (the limitations associated with the local nature of the avoidance methods).

Figure 35.19 shows a snapshot of an experiment carried out with this motion system.

35.12 Conclusions, Future Directions, and Further Reading

The algorithmic tools presented in this chapter show that motion planning and obstacle avoidance research techniques have reached a level of maturity that allow their transfer onto real platforms. Today, several indoor mobile robots use obstacle avoidance techniques on a daily basis to guide visitors in museums. Outdoor applications still require some developments in terms of perception and modeling. For these applications, 3-D sensing capabilities are necessary to model the environment but also to detect obstacles. As an example, several car manufacturer are working on parallel parking assistance. The difficult point of this application is to build a model of a parking spot from 3-D data in a great variety of environments.

The grand challenge of autonomous motion in mobile robots consists today of integrating techniques from different robotics research domains in order to

plan and execute motions for very complex systems such as humanoid robots. Integration in the sense that different software components need to work together on a machine but also in a scientific meaning: the classical motion planning formulation with configuration variables that locate the robot with respect to a global reference frame does not fit partially known environments with imprecise maps. Robotic tasks like motion need to be specified with respect to landmarks of the environment. For instance, grasping an object is by definition a motion specified with respect to the position of the object. Very little has been done to design a general framework in this direction.

Useful complementary readings on motion planning and obstacle avoidance for mobile robots are [35.5, 67–69].

References

- 35.1 N.J. Nilson: A mobile automaton: an application of artificial intelligence techniques, 1st Int. Joint Conf. Artif. Intell. (1969) pp. 509–520
- 35.2 A. Thompson: The navigation system of the JPL robot, 5th Int. Joint Conf. Artif. Intell. (Cambridge 1977) pp. 749–757
- 35.3 G. Giralt, R. Sobek, R. Chatila: A multi-level planning and navigation system for a mobile robot: a first approach to Hilare, 6th Int. Joint Conf. Artif. Intell. (Tokyo 1979) pp. 335–337
- 35.4 T. Lozano-Pérez: Spatial planning: a configuration space approach, IEEE Trans. Comput. **32**(2), 108–120 (1983)
- 35.5 J.C. Latombe: *Robot Motion Planning* (Kluwer Academic, Dordredt 1991)
- 35.6 J.P. Laumond: Feasible trajectories for mobile robots with kinematic and environment constraints. In: *Intelligent Autonomous Systems*, ed. by F.C.A. Groen (L.O. Hertzberger, Amsterdam 1987) pp. 346–354
- 35.7 Z. Li, J.F. Canny: *Nonholonomic Motion Planning* (Kluwer Academic, Dordredt 1992)
- 35.8 H. Sussmann: Lie brackets, real analyticity and geometric control. In: *Differential Geometric Control Theory*, Progress in Mathematics, Vol. 27, ed. by R. Brockett, R. Millman, H. Sussmann (Michigan Technological Univ., Birkhauser 1982) pp. 1–116
- 35.9 H.J. Sussmann, V. Jurdjevic: Controllability of non-linear systems, J. Differ. Equ. **12**, 95–116 (1972)
- 35.10 J.P. Laumond: Singularities and topological aspects in nonholonomic motion planning. In: *Nonholonomic Motion Planning*, Vol. 192, ed. by Z. Li, J.F. Canny (Kluwer Int. Ser. Eng. Comput. Sci., Dordredt 1992)
- 35.11 J.P. Laumond, J.J. Risler: Nonholonomic systems: controllability and complexity, Theor. Comput. Sci. **157**, 101–114 (1996)
- 35.12 J.P. Laumond, P. Jacobs, M. Taïx, R. Murray: A motion planner for nonholonomic mobile robot, IEEE Trans. Robot. Autom. **10**(5), 577–593 (1994)
- 35.13 P. Cheng, S.M. LaValle: Reducing metric sensitivity in randomized trajectory design, IEEE/RSJ Int. Conf. Intell. Robots Syst. (2001) pp. 43–48
- 35.14 J.A. Reeds, R.A. Shepp: Optimal paths for a car that goes both forward and backwards, Pacific J. Math. **145**(2), 367–393 (1990)
- 35.15 P. Souères, J.P. Laumond: Shortest path synthesis for a car-like robot, IEEE Trans. Autom. Contr. **41**(5), 672–688 (1996)
- 35.16 D. Balkcom, M. Mason: Time optimal trajectories for bounded velocity differential drive vehicles, Int. J. Robot. Res. **21**(3), 199–218 (2002)
- 35.17 D. Tilbury, R. Murray, S. Sastry: Trajectory generation for the n -trailer problem using Goursat normal form, IEEE Trans. Autom. Contr. **40**(5), 802–819 (1995)
- 35.18 S. Sekhavat, J.P. Laumond: Topological property for collision-free nonholonomic motion planning: the case of sinusoidal inputs for chained form systems, IEEE Trans. Robot. Autom. **14**(5), 671–680 (1998)
- 35.19 M. Fliess, J. Lévine, P. Martin, P. Rouchon: Flatness and defect of non-linear systems: Introductory theory and examples, Int. J. Contr. **61**(6), 1327–1361 (1995)
- 35.20 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness and motion planning: the car with n trailers, Eur. Contr. Conf. (1993) pp. 1518–1522
- 35.21 F. Lamiriaux, J.P. Laumond: Flatness and small-time controllability of multibody mobile robots: application to motion planning, IEEE Trans. Autom. Contr. **45**(10), 1878–1881 (2000)
- 35.22 P. Rouchon: Necessary condition and genericity of dynamic feedback linearization, J. Math. Syst. Esti. Contr. **4**(2), 1–14 (1994)
- 35.23 P. Rouchon, M. Fliess, J. Lévine, P. Martin: Flatness, motion planning and trailer systems, IEEE Int. Conf. Decision Contr. (San Antonio 1993) pp. 2700–2705
- 35.24 S. Sekhavat, J. Hermosillo: Cycab bi-steerable cars: a new family of differentially flat systems, Adv. Robot. **16**(5), 445–462 (2002)
- 35.25 J. Barraquand, J.C. Latombe: Nonholonomic multibody mobile robots: controllability and motion planning in the presence of obstacles, Algorithmica **10**, 121–155 (1993)
- 35.26 A. Divelbiss, T. Wen: A path space approach to nonholonomic motion planning in the presence of obstacles, IEEE Trans. Robot. Autom. **13**(3), 443–451 (1997)
- 35.27 S. LaValle, J. Kuffner: Randomized kinodynamic planning, IEEE Int. Conf. Robot. Autom. (1999) pp. 473–479
- 35.28 F. Lamiriaux, E. Ferré, E. Vallée: Kinodynamic motion planning: connecting exploration trees using trajectory optimization methods, Int. Conf. Robot. Autom. (2004) pp. 3987–3992
- 35.29 V. Lumelsky, A. Stepanov: Path planning strategies for a point mobile automation moving admist unknown obstacles of arbitrary shape, Algorithmica **2**, 403–430 (1987)
- 35.30 R. Chatila: Path planning and environmental learning in a mobile robot system, Eur. Conf. Artif. Intell. (1982)
- 35.31 L. Strategiesl: Strategies for solving collision-free trajectories problems for mobile robots and manipulator robots, Int. J. Robot. Res., **3**(4), 51–65 (1984)
- 35.32 R. Chattergy: Some heuristics for the navigation of a robot, Int. J. Robot. Res. **4**(1), 59–66 (1985)

- 35.33 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, *Int. J. Robot. Res.* **5**, 90–98 (1986)
- 35.34 B.H. Krogh, C.E. Thorpe: Integrated path planning and dynamic steering control for autonomous vehicles, *IEEE Int. Conf. Robot. Autom.* (1986) pp.1664–1669
- 35.35 J. Borenstein, Y. Koren: Real-time obstacle avoidance for fast mobile robots, *IEEE Trans. Syst. Man Cybern.* **19**(5), 1179–1187 (1989)
- 35.36 K. Azarm, G. Schmidt: Integrated mobile robot motion planning and execution in changing indoor environments, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (1994) pp. 298–305
- 35.37 A. Masoud, S. Masoud, M. Bayoumi: Robot navigation using a pressure generated mechanical stress field, the biharmonic potential approach, *IEEE Int. Conf. Robot. Autom.* (1994) pp.124–129
- 35.38 L. Singh, H. Stephanou, J. Wen: Real-time robot motion control with circulatory fields, *IEEE Int. Conf. Robot. Autom.* (1996) pp.2737–2742
- 35.39 J. Borenstein, Y. Koren: The vector field histogram—fast obstacle avoidance for mobile robots, *IEEE Trans. Robot. Autom.* **7**, 278–288 (1991)
- 35.40 J. Minguez: The obstacle restriction method (ORM): obstacle avoidance in difficult scenarios, *IEEE Int. Conf. Intell. Robot Syst.* (2005)
- 35.41 W. Feiten, R. Bauer, G. Lawitzky: Robust obstacle avoidance in unknown and cramped environments, *IEEE Int. Conf. Robot. Autom.* (1994) pp.2412–2417
- 35.42 D. Fox, W. Burgard, S. Thrun: The dynamic window approach to collision avoidance, *IEEE Robot. Autom. Magaz.* **4**(1), 23–33 (1997)
- 35.43 P. Fiorini, Z. Shiller: Motion planning in dynamic environments using velocity obstacles, *Int. J. Robot. Res.* **17**(7), 760–772 (1998)
- 35.44 R. Simmons: The curvature-velocity method for local obstacle avoidance, *IEEE Int. Conf. Robot. Autom.* (1996) pp.3375–3382
- 35.45 J. Minguez, L. Montano: nearness diagram (ND) navigation: collision avoidance in troublesome scenarios, *IEEE Trans. Robot. Autom.* **20**(1), 45–59 (2004)
- 35.46 J. Minguez, J. Osuna, L. Montano: A divide and conquer strategy to achieve reactive collision avoidance in troublesome scenarios, *IEEE Int. Conf. Robot. Autom.* (2004)
- 35.47 R.B. Tilove: Local obstacle avoidance for mobile robots based on the method of artificial potentials, *IEEE Int. Conf. Robot. Autom.* (1990) pp.566–571
- 35.48 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1999)
- 35.49 J. Minguez, L. Montano: Extending reactive collision avoidance methods to consider any vehicle shape and the kinematics and the dynamic constraints. *IEEE Trans. Robot.* (in press)
- 35.50 J. Minguez, L. Montano, J. Santos-Victor: Abstracting the vehicle shape and kinematic constraints from the obstacle avoidance methods, *Auton. Robots* **20**(1), 43–59 (2006)
- 35.51 J. Minguez, L. Montano: Robot navigation in very complex dense and cluttered indoor/outdoor environments, 15th IFAC World Congress (2002)
- 35.52 A. De Luca, G. Oriolo: Local incremental planning for nonholonomic mobile robots, *IEEE Int. Conf. Robot. Autom.* (1994) pp.104–110
- 35.53 A. Bemporad, A. De Luca, G. Oriolo: Local incremental planning for car-like robot navigating among obstacles, *IEEE Int. Conf. Robot. Autom.* (1996) pp.1205–1211
- 35.54 S. Quinlan, O. Khatib: Elastic bands: Connecting path planning and control, *IEEE Int. Conf. Robot. Autom.* (1993) pp.802–807
- 35.55 O. Brock, O. Khatib: Real-time replanning in high-dimensional configuration spaces using sets of homotopic paths, *IEEE Int. Conf. Robot. Autom.* (2000) pp.550–555
- 35.56 F. Lamiroux, D. Bonnafous, O. Lefebvre: Reactive path deformation for nonholonomic mobile robots, *IEEE Trans. Robot.* **20**(6), 967–977 (2004)
- 35.57 O. Brock, O. Khatib: High-speed navigation using the global dynamic window approach, *IEEE Int. Conf. Robot. Autom.* (1999) pp.341–346
- 35.58 I. Ulrich, J. Borenstein: VFH*: local obstacle avoidance with look-ahead verification, *IEEE Int. Conf. Robot. Autom.* (2000) pp.2505–2511
- 35.59 J. Minguez, L. Montano: Sensor-based motion robot motion generation in unknown, dynamic and troublesome scenarios, *Robot. Auton. Syst.* **52**(4), 290–311 (2005)
- 35.60 C. Stachniss, W. Burgard: An integrated approach to goal-directed obstacle avoidance under dynamic constraints for dynamic environments, *IEEE-RSJ Int. Conf. Intell. Robots Syst.* (2002) pp.508–513
- 35.61 R. Philipsen, R. Siegwart: Smooth and efficient obstacle avoidance for a tour guide robot, *IEEE Int. Conf. Robot. Autom.* (2003)
- 35.62 L. Montesano, J. Minguez, L. Montano: Lessons learned in integration for sensor-based robot navigation systems, *Int. J. Adv. Robotic Syst.* **3**(1), 85–91 (2006)
- 35.63 F. Lu, E. Miliot: Robot pose estimation in unknown environments by matching 2-D range scans, *Intell. Robotic Syst.* **18**, 249–275 (1997)
- 35.64 J. Minguez, L. Montesano, F. Lamiroux: Metric-based iterative closest point scan matching for sensor displacement estimation, *IEEE Trans. Robot.* **22**(5), 1047–1054 (2006)
- 35.65 A. Stenz: The focussed D^* algorithm for real-time replanning, *Int. Joint Conf. Artif. Intell.* (IJCAI) (1995) pp.1652–1659

- 35.66 S. Koenig, M. Likhachev: Improved fast replanning for robot navigation in unknown terrain, Int. Conf. Robot. Autom. (2002)
- 35.67 J.P. Laumond: Robot motion planning and control. In: *Lecture Notes in Control and Information Science*, ed. by J.P. Laumond (Springer, New York 1998)
- 35.68 H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun: *Principles of Robot Motion* (MIT Press, Cambridge 2005)
- 35.69 S.M. LaValle: *Planning Algorithms* (Cambridge Univ., New York 2006)

World Model

36. World Modeling

Wolfram Burgard, Martial Hebert

In this chapter we describe popular ways to represent the environment of a mobile robot. For indoor environments, which are often stored using two-dimensional representations, we discuss occupancy grids, line maps, topological maps, and landmark-based representations. Each of these techniques has its own advantages and disadvantages. Whilst occupancy grid maps allow for quick access and can efficiently be updated, line maps are more compact. Also landmark-based maps can efficiently be updated and maintained, however, they do not readily support navigation tasks such as path planning like topological representations do.

Additionally, we discuss approaches suited for outdoor terrain modeling. In outdoor environments, the flat-surface assumption underling many mapping techniques for indoor environments is no longer valid. A very popular approach in this context are elevation and variants maps, which store the surface of the terrain over a regularly spaced grid. Alternatives to such maps are point clouds, meshes, or three-dimensional grids,

36.1	Historical Overview	854
36.2	World Models for Indoors and Structured Environments	855
36.2.1	Occupancy Grids	855
36.2.2	Line Maps	857
36.2.3	Topological Maps	858
36.2.4	Landmark-Based Maps	859
36.3	World and Terrain Models for Natural Environments	859
36.3.1	Elevation Grids	859
36.3.2	3-D Grids and Point Sets	861
36.3.3	Meshes	862
36.3.4	Cost Maps	863
36.3.5	Semantic Attributes	864
36.3.6	Heterogeneous and Hierarchical Models	866
36.4	Dynamic Environments	866
36.4.1	Conclusion and Further Reading ...	867
	References	867

which provide a greater flexibility but have higher storage demands.

The construction of models of the environment is crucial to the development of several applications of mobile robot systems. It is through these environment models that the robot can adapt its decisions to the current state of the world. The models are constructed from sensor data as the robot discovers its environment. There are three challenges in constructing environment models from sensor data. First, the models must be compact so that they can be used efficiently by other components of the system, such as path planners. Second, the models must be adapted to the task and to the type of environment. For example, modeling the environment

as a set of planes is not relevant to a robot operating in natural terrain. In particular, this implies that a *universal* representation for mobile robots is not possible and that we must instead choose from an array of different approaches. Third, the representation must accommodate the uncertainty inherent to both sensor data and to the robot's state estimation system. The latter point is particularly important since environment models typically accumulate sensor readings over substantial distances into a common reference frame. Drift in position estimates are unavoidable and must be accounted for in the model representation and in its construction.

36.1 Historical Overview

Historically, work in this area concentrated first on robots operating in indoor environments. In that case, the models take advantage of the fact that the world can be represented as vertical structures on reference ground planes. This simplification can be used for representing the world as a two-dimensional (2-D) grid. Uncertainty in the measurements and in the robot's pose can then be modeled by using probabilities of occupancy in the grid rather than binary occupied/empty flags. Another feature of indoor environments is that they are highly structured in that they contain primarily linear structures such as lines and planes. This observations led to a second class of representations based on collections of points, lines, and planes to represent the environment. Here again, a lot of attention was paid to representing the uncertainty on the relative poses of these geometric elements, including a considerable amount of work in the 1980s on using Kalman filters and other probabilistic techniques.

With continued progress on sensing (e.g., longer-range image laser scanners and stereo vision), and on mechanical and controls aspects of mobile robots systems, it became possible to develop mobile robot systems for operations in unstructured, natural terrain, motivated in part by planetary exploration and military applications. In these scenarios, it is no longer appropriate to project the data in a 2-D grid, and environments cannot be described adequately by a small vocabulary of geometric elements. Since, in most cases, it is still appropriate to assume (at least locally) that there is a reference ground plane, a natural representation is a 2-1/2-D grid, in which each cell contains the elevation (and possibly other features) of the terrain at that location. Although they have been used extensively, the main challenges with such *elevation maps* are that they are not compact representations and that it is difficult to incorporate uncertainty in the representation. Accordingly, much of the research has focused on designing efficient data structures and algorithms for elevation maps, for example, hierarchical representations. Recently, the uncertainty challenge has been tackled by using probabilistic representation of the elevation maps.

While elevation maps provide a natural representation for many types of natural terrains, they cannot represent environments with vertical or overhanging structures. This limitation has become more evident in

recent years as applications of mobile robots are increasingly demanding operation in urban environment (in which building walls and other structures cannot be represented by elevation maps) and the use of aerial data, which involves overhanging structures such as tree canopies. This has led to the development of representations that are truly three dimensional (3-D), such as point clouds, 3-D grids, and meshes. The two challenges described above apply here as well, except that the situation is more complicated because of the increased complexity introduced by the third dimension. Current research includes efficient computation over 3-D structures and probabilistic representations of 3-D data.

Because of the large volume of data involved in all of these representations, it is important to be able to group the data into larger chunks corresponding to semantically meaningful parts of the environment. This can be done at different levels, depending on the application the environment. At the lowest level, work in this area involves classifying the points into classes that are relevant to navigation tasks (e.g., distinguishing between vegetation and ground, extracting walls, tree surfaces, etc.). At an intermediate level of representation, this part of the work involves extracting parts of the environments that are considered landmarks of interest for the navigation task (e.g., roads). Finally, the highest representation level involves extracting and representing objects in the environments (e.g., natural obstacles, or specific objects such as cars for operation in urban environments).

All of these representations assume a static environment. In fact, many of the current applications of mobile robots call for operation in mixed environments in which the robot shares its environments with other moving agents, including other robots, people, and vehicles. Assuming that perception algorithms are able to detect and track individual moving objects in the environment, the challenge here is to insert this information into a representation that can be used by a planner. In this case, any one of the previous representations can be used as a foundation, but it needs to be extended to handle the temporal dimension by storing a history of the locations and states of the detected objects. Such representations include the detected trajectories of the detected objects and, in some cases, information on the future predicted trajectories of the objects.

36.2 World Models for Indoors and Structured Environments

36.2.1 Occupancy Grids

Occupancy grid maps, which were introduced in the 1980s by *Moravec* and *Elfes* [36.1], are a popular, probabilistic approach to represent the environment. They are an approximative technique in which we calculate for each cell of a discrete grid the posterior probability that the corresponding area in the environment is occupied by an obstacle. The advantage of occupancy grid maps lies in the fact that they do not rely on any predefined features. Additionally, they offer a constant-time access to grid cells and provide the ability to represent unknown (unobserved) areas, which can be important, for example, in exploration tasks. Their disadvantage lies in potential discretization errors and the high memory requirements.

Throughout this section we assume that the map m consists of a discrete, two-dimensional grid of L cells denoted as m_1, \dots, m_L . Given the sensory input $z_{1:t}$ obtained by the robot at the corresponding positions $x_{1:t}$, the occupancy grid mapping approach calculates a posterior probability $p(m | x_{1:t}, z_{1:t})$.

To keep the calculations tractable, the overall approach assumes that the individual cells of the grid are independent, i.e., that the following equation holds:

$$p(m | x_{1:t}, z_{1:t}) = \prod_{l=1}^L p(m_l | x_{1:t}, z_{1:t}). \quad (36.1)$$

Note that this assumption is rather strong. It basically states that any information about the occupancy of one cell does not tell us anything about its neighboring cells. In practice, we often find objects that are larger than the individual grid cells, like doors, cabinets, chairs etc. Thus, if we know that one cell is occupied, for each of its neighboring cells the probability raises that it is occupied. Despite this fact, occupancy grid maps have been successfully applied in numerous installations of mobile robots and have been proven to be a powerful tool that supports various navigation tasks such as localization and path planning.

Because of the independence assumption expressed by (36.1) we can concentrate on the estimation of the occupancy probability of the individual cells m_l in m . Under additional independence assumptions one can finally arrive at the following formula for calculating the occupancy probability $p(m_l | x_{1:t}, z_{1:t})$ that cell m_l is occupied given $p(m_l | x_{1:t-1}, z_{1:t-1})$ and the new obser-

vation z_t observed at position x_t

$$\begin{aligned} & p(m_l | x_{1:t}, z_{1:t}) \\ &= \left(1 + \frac{(1 - p(m_l | x_t, z_t))}{p(m_l | x_t, z_t)} \cdot \frac{p(m_l)}{(1 - p(m_l))} \right. \\ & \quad \left. \times \frac{1 - p(m_l | x_{1:t-1}, z_{1:t-1})}{p(m_l | x_{1:t-1}, z_{1:t-1})} \right)^{-1}. \end{aligned} \quad (36.2)$$

In practice, one often assumes that the prior $p(m_l)$ is 0.5 so that the second factor in the product becomes 1 and therefore vanishes from the equation.

Additionally, if we define

$$\text{Odds}(x) = \frac{p(x)}{1 - p(x)}, \quad (36.3)$$

the incremental updates can be computed using the following equation:

$$\begin{aligned} & \text{Odds}(m_l | x_{1:t}, z_{1:t}) \\ &= \text{Odds}(m_l | x_t, z_t) \cdot \text{Odds}(m_l)^{-1} \\ & \quad \times \text{Odds}(m_l | x_{1:t-1}, z_{1:t-1}). \end{aligned} \quad (36.4)$$

To recover the occupancy probability from the Odds representation given in (36.4), one can use the following formula, which can easily be derived using (36.3):

$$p(x) = \frac{\text{Odds}(x)}{1 + \text{Odds}(x)}. \quad (36.5)$$

It remains to describe how to compute the occupancy probability $p(m_l | x_t, z_t)$ of a grid cell given a *single* observation z_t and the corresponding pose x_t of the robot. This quantity strongly depends on the sensor of the robot and has to be defined individually for each type of sensor. Additionally, the parameters of these models have to be adapted according to the properties of each sensor. Let us assume that the function $\text{dist}(x_t, m_l)$ refers to the distance between the sensor at pose x_t and the center of the cell m_l . Let us first assume that we only need to consider the optical axis of the sensor cone as is, for example, the case for laser beams sent out by a laser range finder. Then, $p(m_l | x_t, z_t)$ can be formulated as

$$p(m_l | x_t, z_t) = \begin{cases} p_{\text{prior}}, & z_t \text{ is a maximum range reading} \\ p_{\text{prior}}, & m_l \text{ is not covered by } z_t \\ p_{\text{occ}}, & |z_t - \text{dist}(x_t, m_l)| < r/2 \\ p_{\text{free}}, & z_t \geq \text{dist}(x_t, m_l) \end{cases} \quad (36.6)$$

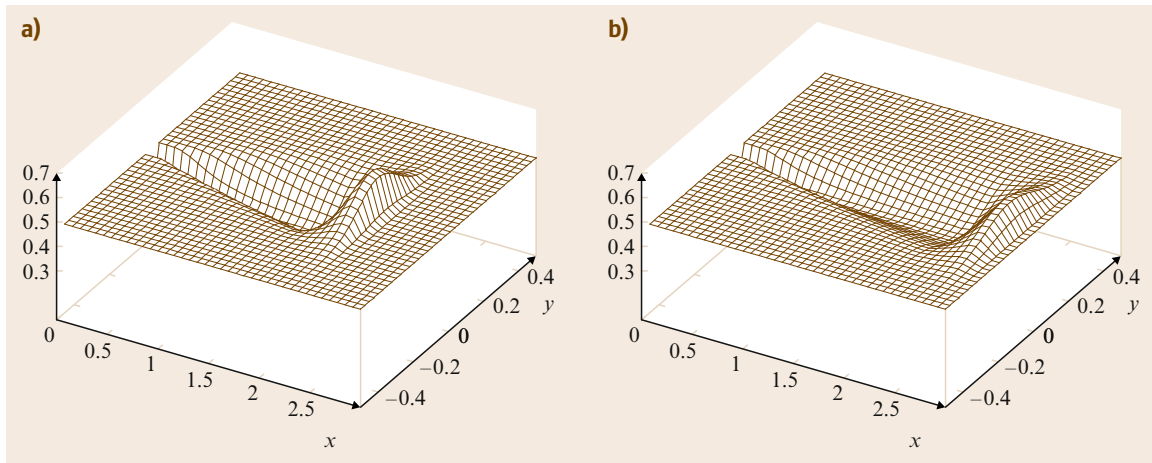


Fig. 36.1a,b Occupancy probability introduced by a single ultrasound measurement of (a) $z = 2.0$ m and (b) $z = 2.5$ m [36.2]



where r is the resolution of the grid map. Obviously, it must hold $0 \leq p_{\text{free}} < p_{\text{prior}} < p_{\text{occ}} \leq 1$.

If a sonar sensor is used, the sensor model is slightly more complicated, since the sensor is not a beam sensor and the observations are more noisy than the ones of a laser range finder. In practice, one typically uses a mixture of three functions to express the model. First, the influence of an observation (which is represented by the difference between p_{prior} and p_{occ} as well as between p_{prior} and p_{free}) decreases with the measured distance. Second, the proximity information of a sonar is substantially affected by noise. Therefore, one typically uses a piecewise-linear function to model a smooth transition from p_{free} to p_{occ} . Finally, the sonar sensor should not be modeled as a beam sensor, since it sends out a conic signal. The accuracy of an observation decreases with the angular distance between the cell under consideration and the optical axis of the observation. This is expressed by the derivation from the prior and is typically modeled using a Gaussian with zero mean. Therefore, it is maximal along the optical axis and decreases the bigger the angular distance from the optical axis is [36.2].

Two examples for a resulting model are depicted in Fig. 36.1, which shows two three-dimensional plots of the resulting occupancy probabilities for a meas-

Fig. 36.2 Incremental mapping in a corridor environment. The *upper left* image shows the initial map and the lower one shows the resulting map. The maps in between are the local maps built from the individual ultrasound scans perceived by the robot [36.2] ◀



Fig. 36.3 Occupancy grid map obtained from ultrasound data [36.2]

urement of 2 m (Fig. 36.1a) and 2.5 m (Fig. 36.1b). In this figure, the optical axis of the sensor cone was identical with the x -axis and the sensor was placed in the origin of the coordinate frame. As can be seen, the occupancy probability is high for cells whose distance to x_t is close to z_t . It decreases for cells with shorter distance than z_t as well as with increasing values of the angular distance.

Figure 36.2 depicts the mapping process for a sequence of observations recorded with an iRobot B21r robot. The first row shows how a map was built from a sequence of previous ultrasound scans. Afterwards the robot perceived a series of 18 ultrasound scans, each consisting of 24 measurements. The occupancy probabilities for these 18 scans are depicted in rows 2–7. The occupancy probability grid obtained by integrating the individual observations into the map is shown in the last row of this figure. As can be seen, the belief converges to a representation of the corridor structure in which the scans were recorded. A typical resulting map obtained for an indoor environment is depicted in Fig. 36.3.

36.2.2 Line Maps

The representation of the environment by line models is a popular alternative to the grid-based approximations described above. Line models have several advantages over these nonparametric representations. They require substantially less memory than grids and therefore scale better with the size of the environment. They furthermore are more accurate since they do not suffer from discretization problems. In this section, we consider the problem of calculating lines that provide the approximation of a set of range points by a line. If the data points are given by n pairs (x_i, y_i) of Cartesian coordinates, the

line that minimizes the squared distances to all points can be calculated in closed form according to

$$\tan 2\phi = \frac{-2 \sum_i (\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i [(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]}, \quad (36.7)$$

$$r = \bar{x} \cos \phi + \bar{y} \sin \phi, \quad (36.8)$$

where $\bar{x} = \frac{1}{n} \sum_i x_i$ and $\bar{y} = \frac{1}{n} \sum_i y_i$. In these equations, r is the normal distance of the line from the origin and ϕ is angle of the normal.

Unfortunately, there is no closed-form solution for situations in which the data points are generated by multiple linear structures. In such a situation two problems arise. First one has to answer the question how many lines there are, and second one has to solve a data association problem, which is to find the assignment of the data points to the individual lines. Once the number of lines and the associations are known, we can apply (36.7) and (36.8) to calculate the individual line parameters.

There is a popular approach to solve this problem for range scans with multiple linear structures. This approach, which goes back to the work of *Douglas and Peucker* [36.3], is also known as the *split-and-merge* algorithm. Its key idea is to recursively subdivide the point set into subsets that can be more accurately approximated by a line. The approach starts with the line calculated from all data points and determines the data point with maximum distance from this line. If this distance is below a given threshold, the algorithm terminates and provides the fitted line as output. Otherwise,

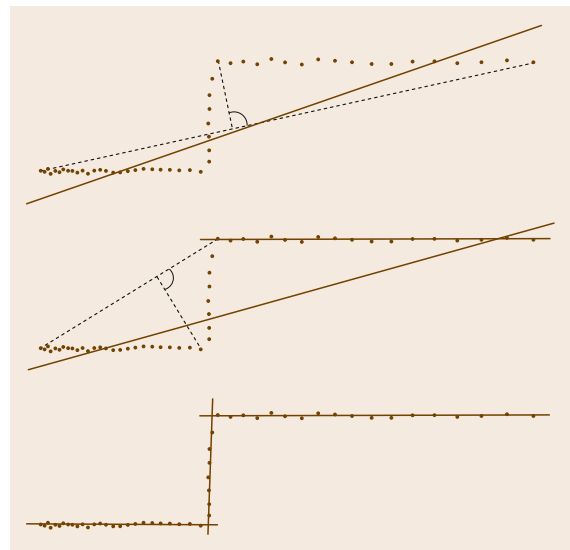


Fig. 36.4 Split-and-merge algorithm applied to a point set

it calculates the point with the largest distance from the line that connects the first and the last point. This point is the so-called splitting point. It then divides the point set into two subsets, one containing all points up to the first to the splitting point and one containing all points after the splitting points. The algorithm is then recursively applied to the two subsets.

Whereas the split-and-merge algorithm is quite effective and efficient, it is not guaranteed that the resulting model is actually optimal, i. e., is the model that minimizes the squared distances of all data points. One approach that is able to find such a model is based on the expectation maximization (EM) algorithm, which in the application described here can be regarded as a variant of the fuzzy k -means clustering algorithm. Let us suppose that the number m of lines in the model θ is known. Let us furthermore suppose that the likelihood of a data point $z = (x, y)$ given the model θ consisting of the line set $\{\theta_1, \dots, \theta_m\}$ is defined as

$$p(z | \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2} \frac{d(z, \theta_j)^2}{\sigma^2}\right), \quad (36.9)$$

where σ is the standard deviation of the measurement noise and θ_j is the line for which Euclidean distance $d(z, \theta_j)$ to z is minimal.

The goal of the EM algorithm is to generate an iterated sequence of models of increased likelihood. To achieve this, one introduces so-called correspondence variables $c_{ij} \in \{0, 1\}$ that specify to which linear component of the model each point belongs. Since the correct values of these assignment variables are unknown, one estimates a posterior about their values. Let θ_j be a component of the model and z_i be a measurement. Then the expectation about c_{ij} , i. e., that measurement i belongs to line j is computed in the E-step as

$$E[c_{ij} | \theta_j, z_i] = p(c_{ij} | \theta_j, z_i) \quad (36.10)$$

$$= \alpha p(z_i | c_{ij}, \theta_j) p(c_{ij} | \theta_j) \quad (36.11)$$

$$= \alpha' p(z_i | \theta_j). \quad (36.12)$$

In the M-step, the algorithm then computes the parameters of the model by taking into account the expectations computed in the E-step:

$$\theta_j^* = \arg \min_{\theta_j'} \sum_i \sum_j E[c_{ij} | \theta_j, z_i] d^2(z_i, \theta_j'). \quad (36.13)$$

Given a fixed variance σ for all data points in z we can determine the most likely model in closed form according to the following equations, which are probabilistic variants of (36.7) and (36.8) that take into account the

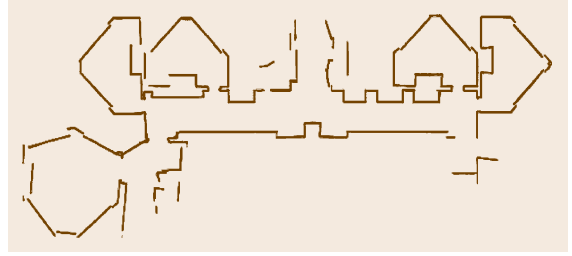


Fig. 36.5 Line map consisting of 94 lines generated for 311 823 range points generated with the EM-based approach [36.4]

data association uncertainty given by the expectations calculated in the E-step:

$$\tan 2\phi_j = \frac{-2 \sum_i E[c_{ij} | \theta_j, z_i] (\bar{x} - x_i)(\bar{y} - y_i)}{\sum_i E[c_{ij} | \theta_j, z_i] [(\bar{y} - y_i)^2 - (\bar{x} - x_i)^2]}, \quad (36.14)$$

$$r_j = \bar{x} \cos \phi_j + \bar{y} \sin \phi_j. \quad (36.15)$$

Here \bar{x} and \bar{y} are computed as

$$\bar{x} = \frac{\sum_i E[c_{ij} | \theta_j, z_i] x_i}{\sum_i E[c_{ij} | \theta_j, z_i]}, \quad \bar{y} = \frac{\sum_i E[c_{ij} | \theta_j, z_i] y_i}{\sum_i E[c_{ij} | \theta_j, z_i]}. \quad (36.16)$$

Figure 36.5 depicts a line map extracted from 311 823 data points using the EM-based approach. In this example, the model consists of 94 lines. One approach to determine the optimal line is to utilize the Bayesian information criterion [36.4].

36.2.3 Topological Maps

In contrast to the previously discussed representations, which mainly focus on the geometric structure of the environment, topological representations have also received significant attraction. One of the pioneering



Fig. 36.6 Example of a generalized Voronoi graph [36.5]



Fig. 36.7 (a) Mobile robot mapping a set of stone rocks. The right image (b) depicts the path and the estimated landmark positions. The manually determined positions of the rocks are marked by circles [36.8]

approaches to topological mapping have been presented in 1988 is the work by *Kuipers* and *Byun* [36.6]. In this approach, the environment is represented by a graph-like structure, in which the nodes are locally distinguishable places and the nodes are travel edges along which the robot can move between the places. Here, distinctive places are identified according to the distance to nearby objects. The distinctive places were defined by *Choset* and colleagues [36.7] as the meet-points in the generalized Voronoi diagrams, i. e., the points with an out-degree of three or more. A generalized Voronoi diagram is the set of points equidistant from the closest two or more obstacle boundaries. Generalized Voronoi diagrams are a very popular representation as they can be considered as road-maps with a high correspondence to the topological structure of the environment. They have been used extensively for path planning. To plan a path from a starting position to a goal point in the environment, all the robot has to do is to first plan a path to the generalized Voronoi graph, then along the generalized Voronoi graph, and then from the generalized Voronoi graph to the goal point [36.7]. Figure 36.6 shows an example generalized Voronoi graph for an indoor environment. Note that in this figure only those parts of the graph that can be traversed by the robot without colliding with objects are displayed.

36.2.4 Landmark-Based Maps

For environments with locally distinguishable features, landmark-based maps have been extensively used. If we assume that the position of the robot is always known, it simply remains to maintain an estimate about the positions of the individual landmarks over time. In a planar environment, m is represented by K two-dimensional Gaussians with mean μ_k and covariance Σ_k , one for each landmark. If a linearized version of the perception problem is given, the individual Gaussians can be updated using the equations for the extended Kalman filter. Note that, compared to the use of the *EKF* for landmark-based simultaneous localization and mapping (*SLAM*), where we needed a $2K + 3$ -dimensional state vector (3 dimensions for the position of the robot and $2K$ dimensions for the positions of the individual landmarks), we only need K two-dimensional Gaussians to represent the entire map since the robot pose is known. This property, for example, has been utilized in the *FastSLAM* algorithm [36.8].

The left image of Fig. 36.7 shows a robot equipped with a *SICK* laser range finder that maps the position of rocks. The right image depicts the path of the vehicle as well as the manually determined and automatically estimated landmark positions.

36.3 World and Terrain Models for Natural Environments

From a survey of environment modeling [36.9] with an emphasis on probabilistic techniques for indoor environments, a taxonomy can be created along several axis: metric versus topological versus semantic, robot-centric versus world-centric, or application-based. We choose to first review purely geometric models (elevation grid, 3-D grid, meshes), then geometric models with low-level attributes (cost maps), and then models with

richer semantic attributes, to finish with heterogeneous and hierarchical models.

36.3.1 Elevation Grids

Assuming that the terrain can be represented as a function $h = f(x, y)$, where x and y are the coordinates on a reference place and h is the corresponding eleva-

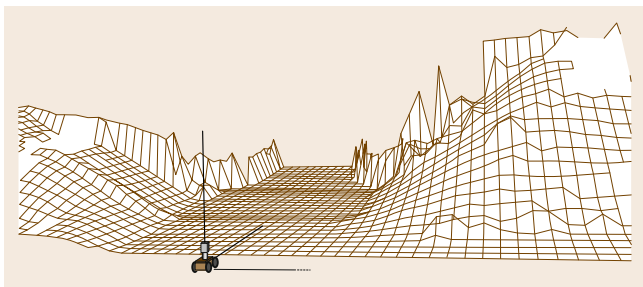


Fig. 36.8 Example elevation map built by accumulating 3-D data from a range sensor

tion. A natural representation is a digital elevation map which stores the value of h at discrete locations (x_i, y_i) . Because they are simple data structures and can be generated from sensor data in a relatively straightforward manner, elevation maps have been used extensively for mobile robots operating in natural environments with no vertical surfaces or overhangs (e.g., for planetary exploration scenarios [36.10]). Several issues need to be addressed when using elevation maps for mobile robots.

A regularly sampled grid is appropriate when the sensor data is roughly uniformly distributed on the reference plane. This is the case, for example, with aerial data. For ground robots, however, the distribution of the data on the reference ground plane varies dramatically because of the small incidence angles with the reference plane. This can be addressed by using variable-size cells instead of regularly sampled cells. In that case, the distribution of the cells on the reference plane is designed to approximate the distribution of points that would be measured by the sensor on this reference plane. Such nonuniform representations are useful primarily in cases in which the map is referenced to the current position

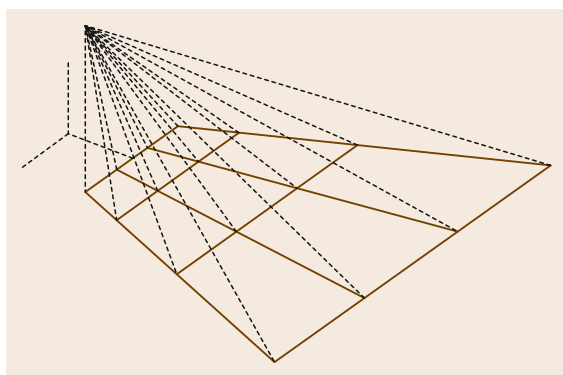


Fig. 36.9 Example of a robot-centric map with variable size (after [36.11])

of the robot [36.11] (Fig. 36.9). Frequent resampling is needed if the map is expressed in a global reference frame.

Irrespective of the sampling scheme used to construct the reference grid, the density of data in the grid varies due to local self-occlusions of the terrain surface and to mismatches between the resolution of the projected sensor data and the resolution of the grid. This can be problematic because, from the point of view of a planner, the elevation map would appear as a scattering of cells with elevation data and cells with no evidence. Various schemes have been proposed to remedy this problem. The basic idea is to estimate the elevation values of the empty cells by interpolating across the cells with known elevation values. This has to be done with great care to avoid filling up regions that correspond to parts of the environment occluded by the terrain (*range shadows*) in which elevation data should not be inferred. Such mistakes can be catastrophic since a planner may generate paths through areas that are entirely unknown. One general approach to this problem is to use surface interpolation techniques including a term that allows for discontinuities in the resulting surfaces [36.12]. An alternative approach is to use the visibility constraints induced by the known geometry of the sensor in order to estimate plausible values of elevation at each cell [36.13, 14].

A difficulty in interpolating techniques is to explicitly take into account the sensor uncertainty. In particular, it is difficult to account for the varying resolution of the sensor as a function of range in a fixed-resolution grid. An alternative is to use multiple grids at different resolutions. The appropriate resolution to be used at a particular point (x, y) can be decided based on the range from (x, y) to the closest sensor location to retrieve the value at (x, y) from the appropriate map for that resolution. Generally, the longer the range, the coarser maps that are used. This approach eliminates the gaps in the map due to undersampling at long range from the sensor by using the optimal resolution based on sensor geometry [36.15].

Uncertainty in the sensor measurement is expressed most naturally with respect to the sensor frame, for example, by expressing the uncertainty in the direction of measurement. As a result, converting such uncertainty models to elevation maps is difficult because a distribution in the direction of the measurement maps to a distribution in the reference plane, not a distribution on the elevation value at a particular grid point. When updating a cell based on sensory input, one has to take into account that the uncertainty in a measurement increases

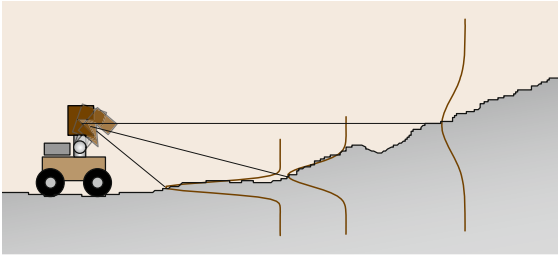


Fig. 36.10 The standard deviation of a height measurements can be modeled to depend linearly on the distance to the grid cell

with the distance measured due to errors in the tilting angle. A popular approach to estimate the height h at location (x, y) is to use a Kalman filter. If we assume that σ is the standard deviation of the current measurement h in the vertical direction in (x, y) and σ_{t-1} is the standard deviation of the current estimate or h_{t-1} , we can apply the following equations to obtain the new estimate h_t with standard deviation σ_t ,

$$h_t = \frac{\sigma^2 h_{t-1} + \sigma_{t-1}^2 h}{\sigma_{t-1}^2 + \sigma^2}, \quad (36.17)$$

$$\sigma_t^2 = \frac{\sigma_{t-1}^2 \sigma^2}{\sigma_{t-1}^2 + \sigma^2}. \quad (36.18)$$

One possible solution to deal with varying uncertainties is to use a model in which the standard deviation of the height of a measurement increases linearly with the length of the range measurement, as indicated in Fig. 36.10.

Vegetation cover is another source of error in the recovery of terrain elevation by totally or partially occluding the ground surface from the vehicle sensors. Online learning techniques can address this issue by predicting ground elevation ahead of the vehicle based on past appearance observations of similar terrain and vehicle state over the terrain [36.17].

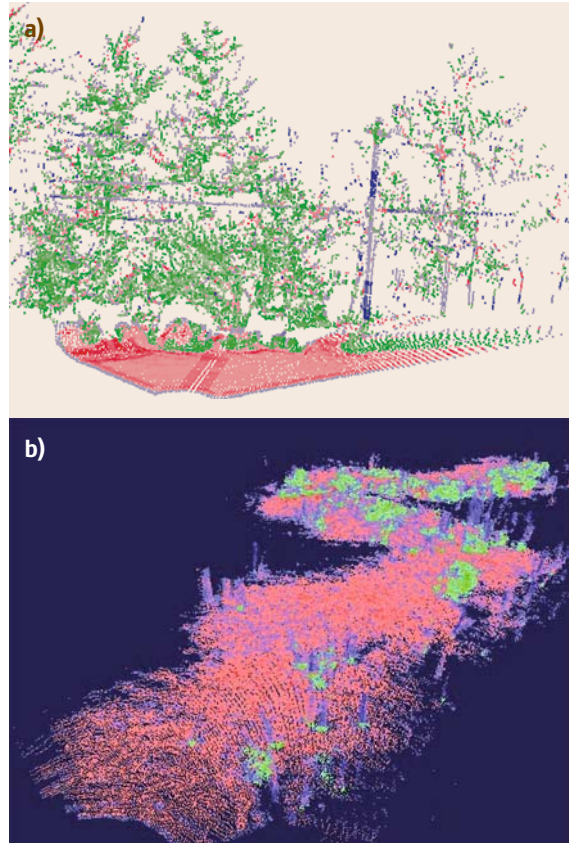


Fig. 36.12a,b Example of mapping and classification of 3-D point sets: (a) 3-D data and classification result (green = vegetation, red = surfaces, blue = lines; the saturation of the color is proportional to the confidence in the classification result); (b) map accumulated over a large number of scans

Intermediate representations between 2-D elevation maps and full 3-D representations, addressed below, are extended elevation maps [36.18] and so-called multilevel surface maps [36.16]. Such approaches are

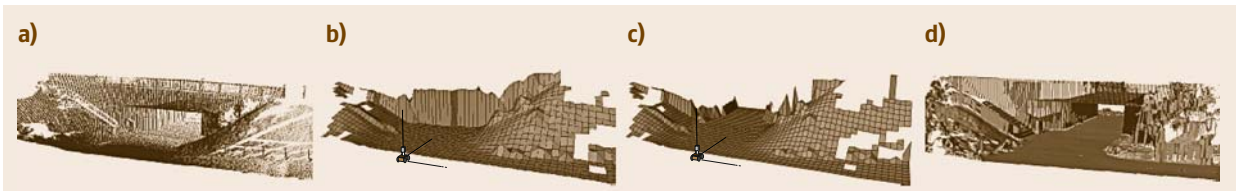


Fig. 36.11a-d Different versions of digital elevation maps [36.16]: (a) scan (point set) of a bridge, (b) a standard elevation map computed from this data set, (c) an extended elevation map, which correctly represents the underpass under the bridge, and (d) a multilevel surface map that correctly represents the height of the vertical objects

specifically useful in structured terrain with vertical objects or overhangs such as bridges (Fig. 36.11).

36.3.2 3-D Grids and Point Sets

Elevation maps as described above assume a reference direction. In many cases, this assumption is violated. An alternative is to represent the data directly in 3-D without projecting it onto a reference 2-D plane. The advantage of doing this is that all the sensor data can be preserved in its original distribution and that there is no restriction on the geometry of the environment. The problem is to be able to manipulate efficiently very large sets of 3-D points. Data structures based on dynamic 3-D grids can be used for this purpose [36.21]. An interesting characteristic of these representations is that they enable the computation of cost evaluated from the true local 3-D distribution of the data (as opposed to costs computed from a local surface. This is important, for example, in environments with vegetation scatter, which cannot be modeled as a surface.

In the past, octree-based 3-D representations have been successfully used to map an underwater cave from several sonars mounted on an underwater vehicle [36.19]. The data structure is designed to maintain efficiently, in terms of memory and access time, hundreds of maps to store particles for localization. However, dense 3-D occupancy grids have also been maintained to model natural environments using a radar [36.20]. An example of each approach is presented in Fig. 36.13.

In the approaches mentioned above, the data sets are accumulated, sometimes probabilistically, into discrete environment volumes. A different approach is to represent the environment with discrete sensor samples

integrated probabilistically into a metric map [36.22]. Such an approach is different from occupancy grids in several aspects: the storage requirement and resolution is adaptive instead of fixed and the potential for extensibility and scalability is higher.

36.3.3 Meshes

As described above, elevation maps are compact and easy to implement but they are restricted to a particular class of terrain; at the other extreme, using 3-D representations directly is more general but it is also more expensive computationally, and it does not represent explicitly surface continuity. A compromise is to represent the map as a mesh. This approach is attractive because it can, in principle, represent any combination of surfaces. It is also a compact representation in that, even though the size of the mesh may be initially very large, efficient mesh simplification algorithms exist in the literature [36.23] and can be used to reduce the map to a small number of vertices.

The key issue with meshes is that extraction of the correct surfaces from raw data can be difficult in complex environments. In particular, the data is corrupted by sensor noise and by random clutter from other sources, such as vegetation, which cannot be represented as a continuous surface. In addition, it is necessary to detect discontinuities in the data precisely so that disconnected pieces of surface do not become accidentally linked in the mesh formation process [36.18]. Figure 36.14 shows typical mesh representations of an urban environment.

In some applications, the objective is to produce terrain models not to support autonomous navigation but to produce a model to be visualized to a human. City planning is a typical application. Examples

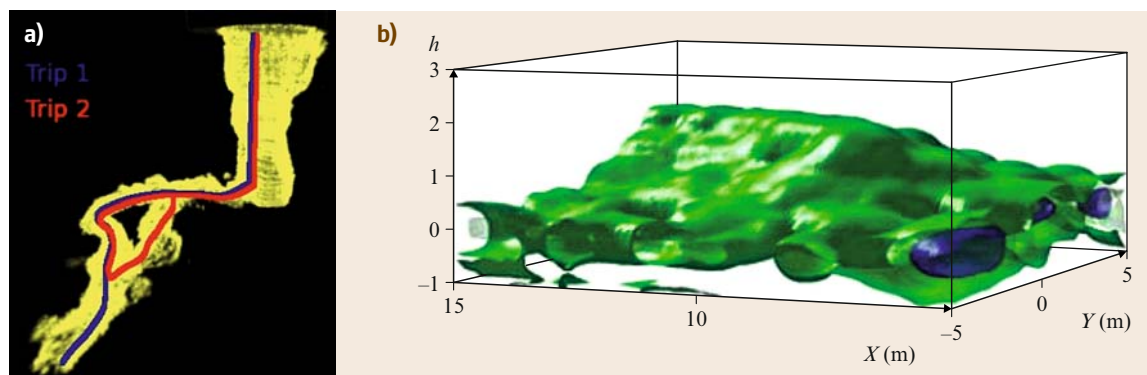


Fig. 36.13a,b Examples of volumetric maps for a natural environment. Whereas (a) shows a map of an underwater cave [36.19], (b) depicts a terrain map obtained from a radar [36.20]



Fig. 36.14a–c Two views (b–c) of the mesh representation of a map (a) obtained by scanning an environment. The meshes are reduced by a factor of 10 from the initial data sampling



Fig. 36.15a–c Examples of urban terrain modeled as a textured mesh ((a) from [36.24], (b) from [36.26], and (c) from [36.25])

include producing 3-D texture models of urban environments using videos from a camera mounted on a moving vehicle equipped with an inertial navigation system [36.24], using two laser scanners in push-broom mode, one for mapping and the other for localization [36.25], with a camera coregistered with the mapping laser, and collecting high-resolution laser data and imagery to produce geometric and photometric cor-

rect 3-D models of buildings [36.26]. Figure 36.15 depicts examples of textured meshes representing urban terrain.

36.3.4 Cost Maps

The most direct use of elevation maps is to compute traversability costs at each cell of the grid. The costs

are computed by comparing the local terrain shape with a kinematic model of the robot. Several approaches have been proposed for computing the costs depending on the exact vehicle model; for example, a cost can be computed by considering the local slope and 3-D texture of the terrain [36.27]. More involved models take into account a detailed model of the robot dynamics. In that case, different costs are generated for different possible robot speeds and path curvatures. The cost grid is used in a minimum-cost planner. Since the terrain map is updated continuously as the robot traverses the environment, it is also necessary to update the costs continuously as new data arrives. Consequently, it is important that the planner is able to support changing costs without needing to process the entire grid every time new data is inserted.

An example of such a combination of grid representation and dynamic planner is the D* system, which uses a version of A* that fully supports dynamic grid updates [36.29–31]. Whenever a grid cell (or a group of grid cells) is updated, the planner makes minimal updates to its internal representation so that the optimal path can be updated quickly.

Defining the exact relation between the costs and the elevation values stored in the grid can be quite difficult. In fact, beyond the limiting cases of terrain with large elevation gradient or large slope, there is little guidance as to why a particular part of the terrain may be more traversable than another as it depends critically on the exact configuration of the robot. For this reason, recent work has focused on deriving cost maps directly from observations, rather than by using handcrafted algorithms. One approach involves learning the best weights to combine a set of predefined costs. Another approach uses costs computed from other sources for inferring how to determine the costs on the terrain currently seen

by the robot; for example, data from overhead imagery can be used to predict the traversability costs that should be used for a ground robot [36.28] or for learning to map local elevation distributions to costs values by analyzing actual robot trajectories through the terrain. Similar online learning approaches are used to predict terrain roughness [36.32], terrain slippage [36.33], or traversability in general [36.34, 35] to be used in a cost map.

36.3.5 Semantic Attributes

The representations described above are concerned only with storing the data in a way that is compact and that can be used to estimate drivability costs. Often it is necessary to reason with high-level knowledge about the environment, e.g., the location and types of objects around the robot or the type of terrain (vegetation, mud, wall) in the environment. For convenience we will refer to this type of information as *semantic attributes* attached to different parts of the environment. There are several different ways to approach the problem of generating and representing semantic attributes. One possible direction is to approach the problem as one of extracting *landmarks* from the environment. As for their indoor counterparts, outdoor environments can also be modeled using landmarks. Landmarks are defined broadly as scene elements easy to detect, salient in their surroundings, and easy to recognize. They can be specific objects, such as rocks [36.36], tree trunks in forest [36.37], or locations with a distinct appearance signature in 2-D or 3-D, for urban terrain [36.38, 39] or natural environments [36.40]. Landmarks can be used to produce topological representations of the environment or used in conjunction with metric maps. Recent efforts involve learning a compact representation of environments without defining a priori what a landmark should be [36.41, 42] or using statistical learning techniques [36.43] to extract new features from images. Nonlinear dimension-reduction techniques are used to obtain the representation and the results were demonstrated onboard ground and aerial vehicles.

Another view of the problem is as a *classification and grouping* problem. Indeed, the map representations described in the earlier sections are low-level representations in that they do not attempt to group the data points into larger structures that are coherent with respect to geometry, terrain type, or semantic content. In practice, one would like to abstract the data into larger units that can be used by a planner or transmitted to another robot or an operator; for example, in an urban environment, one would like to group the parts of the data that belong

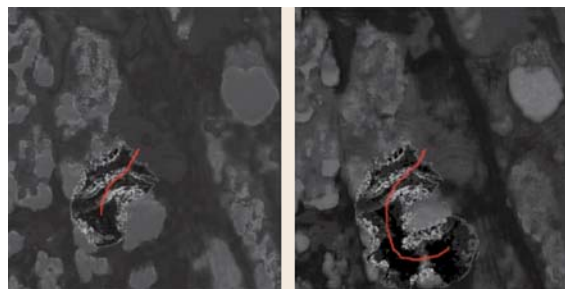


Fig. 36.16 Example of online learning of cost [36.28]. The vehicle trajectory is represented in red. As the vehicle progresses, note the change in cost over all the map. The darker the appearance, the lower the cost

to planes corresponding to pieces of walls. One such example is presented in Fig. 36.18.

One approach is to group 3-D points into components based on low-level classification and feature detection [36.45]. The shape of these component is analyzed further to discriminate between various natural object components (tree trunks, branches) and manmade obstacle (wires). Geometric primitives are then fitted to the components (mesh for the ground surface, ball tree for the vegetation, and cylinder to branches) to produce a compact high-level geometric description of the terrain.

Local terrain classification can be achieved by computing local features from the map and classifying each element of the map in different classes of terrain. This can be done from an elevation map, in which case the *elements* are the cells of the elevation map, or from a point cloud representation in which case the elements are 3-D locations. Given a location in the map x , a feature vector is $V(x)$ is computed in a neighborhood $N(x)$ of x and a classifier $f(V)$ returns the type of terrain at x . Features that have been used in the past are statistics of the distribution of the map data around each element, such as the slope and distribution of elevation [36.46], and the second-order moment of the distribution of the 3-D points in a neighborhood [36.45]. The terrain classes depend on the application. The most direct classification approach uses a binary classifier that separates the terrain into obstacle regions and traversable regions. More involved classifiers segment the map into more classes such as vegetation, solid surfaces, and linear structures [36.13, 45, 47]; an example is shown in Fig. 36.12. In some cases, it is possible to store, with each data element x in the map, the direction in which a measurement was taken for this element. In this case, it is possible to refine the classification by reasoning about the intersection of the measurement ray $d(x)$ with the rest of map; for example, this type of geometric reasoning [36.48] has been exploited to recover the load-bearing surface obscured by vegetation [36.40, 49, 50] and for extracting *negative* obstacles (such as ditches) [36.51]. In all of these cases, the classification information cannot be recovered directly from local statistics and must be inferred from longer-range geometric reasoning.

This class of approaches to local feature classification is similar to approaches for extracting features from images and it suffers from similar limitations. Specifically, these representations are sensitive to the choice of neighborhood used for computing the features. If it is too large, information over a large area is aver-

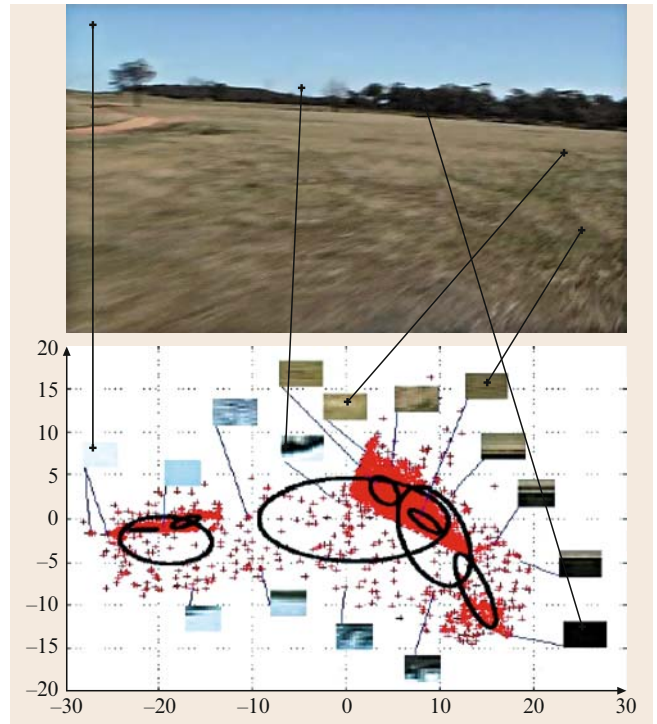


Fig. 36.17 Sample image and low-dimensional embedding of randomly sampled high-dimensional image patches [36.42]

aged out, leading to poor classification performance. If it is too small, there is not enough information in the neighborhood for reliable classification. The situation is complicated by the fact that the resolution of the map, or more precisely the density of data points in the map, may vary drastically as the distance from the sensor changes. This problem is addressed by using different neighbor-

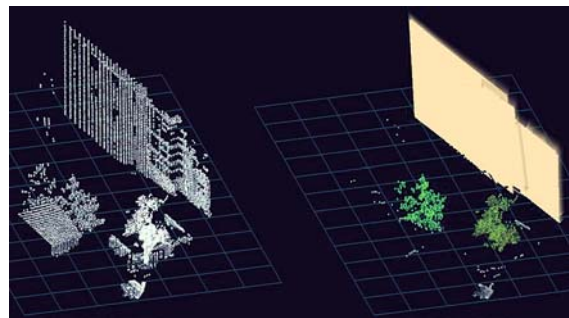


Fig. 36.18 Terrain classification and extraction of geometric features. Whereas the 3-D data is shown on the *left*, the feature map with extracted planes and vegetation regions is shown on the *right* [36.44]

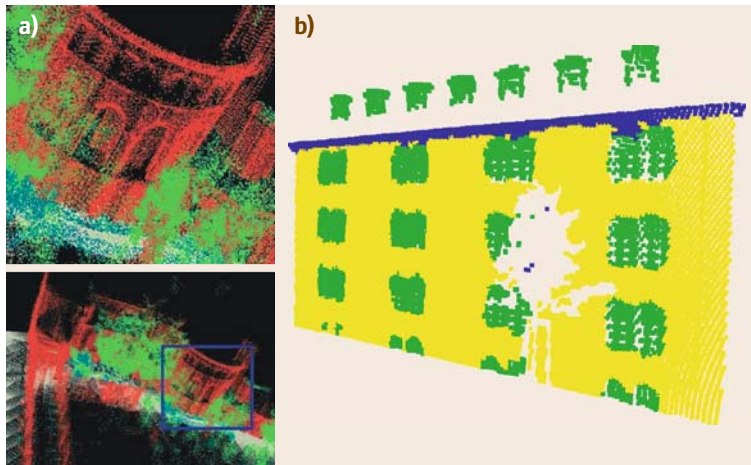


Fig. 36.19 (a) Structure learning for terrain classification [36.54] and (b) building features extraction [36.55]

hood sizes at different locations in the map, or by tuning the classifier differently depending on the map location, typically based on distance from the sensor [36.15, 52]. A second problem is to generate the classifier f . This can be done by using a physical model that predicts the statistics of the local data distribution in the map assuming different terrain types [36.53]. This is generally difficult and a preferred approach is to train the classifier on training data.

This level of classification provides information about the local type of terrain, which can be used in planning, but it does not extract the extended geometric structures that may exist in the environment (Fig. 36.18). Geometric structures such as planar patches can in principle be extracted by using techniques similar to the ones described in the context of indoor environments, such as EM, with the added difficulty that there is a larger amount of clutter that complicates the extraction of the patches. Robust techniques that can handle this level of clutter are typically used for extracting the planes [36.56, 57].

Classification based on local attributes can be improved significantly by taking into account contextual information; for example, hidden Markov models have been successfully used for laser data analysis to determine terrain traversability [36.58]. Another structure learning approach based on Markov random fields in conjunction with margin-maximization

criterion has been demonstrated for a wide class of applications, including 3-D terrain classification and object segmentation [36.54] or building structures extraction [36.55] (Fig. 36.19). Finally there are approaches to detect, select, model, and recognize natural landmarks automatically for robot localization and environment mapping [36.36].

36.3.6 Heterogeneous and Hierarchical Models

For long-range navigation, an autonomous vehicle must perform numerous tasks, including absolute and relative localization, path planning, and reactive obstacle avoidance. In addition, it must perform some tasks to fulfill the mission requirements, detecting and modeling objects, for example. To achieve this, a hierarchical framework has been presented that accounts for the different scales and granularities of the representation needed [36.46]. Furthermore, models build from heterogeneous imagery sources (overhead, descent, and ground) have been used to produce 3-D multiresolution terrain models that support Mars exploration [36.59]. The hybrid metric map [36.60], enhances feature maps with metric maps to provide a dense but compact environment representation. Another representation extracts landmarks based on laser/imagery appearance as well as a metric map [36.61].

36.4 Dynamic Environments

The majority of techniques for mapping have been developed for static environments. Certain approaches

like occupancy grids or elevation maps can in principle deal with dynamic environments in which objects



Fig. 36.20 (a) A mobile robot acquiring a three-dimensional scan of an urban scene. (b) The people in the scene cause spurious data points in the resulting meshes. (c) The same scene after filtering people

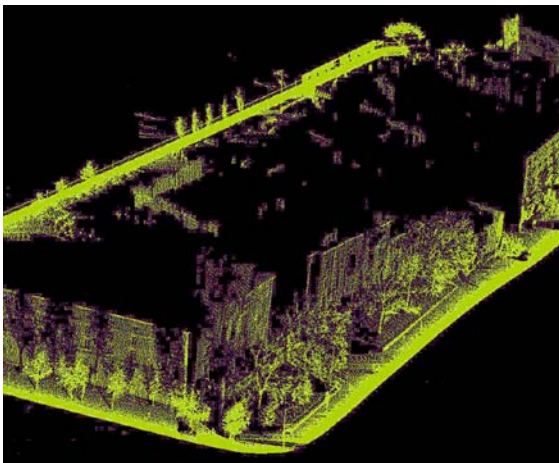


Fig. 36.21 Complex three-dimensional scene acquired with a mobile robot after filtering dynamic objects 36.63

move. Their drawback lies in the fact that the time to unlearn that a cell is free or that the elevation has changed can require as many observations as the robot received with the same area being occupied or with a different elevation. To resolve this

problem, several alternative approaches have been proposed in the past. One very popular technique is to track moving objects using feature-based tracking algorithms [36.62, 63]. Such approaches are especially useful when the type of the dynamic objects is known in advance. They have been successfully applied in the context of learning three-dimensional city maps from range data. Figures 36.20 and 36.21 show applications in which dynamic objects were removed from the 3-D data acquired with mobile robots in urban scenes. Alternative approaches to such tracking techniques include those that learn maps on different time scales [36.64], that explicitly learn different states of dynamic environments [36.65], or that only map the static aspects [36.66].

36.4.1 Conclusion and Further Reading

Further reading about typical representations and how they can be utilized for mobile robot navigation can be found in recent textbooks on mobile robotics [36.2, 67, 68]. A comprehensive survey of the fundamentals of spatial data structures and their applications is available in *Samet* [36.69].

References

- 36.1 H.P. Moravec, A.E. Elfes: High resolution maps from wide angle sonar, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (1985)
- 36.2 H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun: *Principles of Robot Motion: Theory, Algorithms and Implementation* (MIT Press, Cambridge 2005)
- 36.3 D.H. Douglas, T.K. Peucker: Algorithms for the reduction of the number of points required to represent a line or its caricature, Cdn. Cartogr. **10**(2), 112–122 (1973)
- 36.4 D. Sack, W. Burgard: A comparison of methods for line extraction from range data, Proc. IVAC Symp. Intell. Auton. Vehicles (IAV) (2004)
- 36.5 P. Beeson, N.K. Jong, B. Kuipers: Towards autonomous topological place detection using the extended Voronoi graph, IEEE Int. Conf. Robot. Autom. (ICRA) (2005)
- 36.6 B.J. Kuipers, Y.-T. Byun: A robust qualitative method for spatial learning in unknown environments, Proc. Nat. Conf. Artif. Intell. (AAAI) (1988)

- 36.7 H. Choset, K. Nagatani: Topological simultaneous localization and mapping (SLAM): toward exact localization without explicit localization, *IEEE Trans. Robot. Autom.* **17**(2), 125–137 (2001)
- 36.8 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: a factored solution to the simultaneous localization and mapping problem, *Proc. Nat. Conf. Artif. Intell. (AAAI)* (2002)
- 36.9 S. Thrun: Robotic mapping: a survey. In: *Exploring Artificial Intelligence in the New Millenium*, ed. by G. Lakemeyer, B. Nebel (Morgan Kaufmann, New York 2002)
- 36.10 M. Maimone, P. Leger, J. Biesiadecki: Overview of the Mars exploration rovers' autonomous mobility and vision capabilities, *IEEE Int. Conf. Robot. Autom.* (2007)
- 36.11 S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, R. Chatila: Autonomous rover navigation on unknown terrains: functions and integration, *Int. J. Robot. Res.* **21**(10–11), 917–942 (2002)
- 36.12 R. Olea: *Geostatistics for Engineers and Earth Scientists* (Kluwer Academic, Dordrecht 1999)
- 36.13 A. Kelly, A. Stentz, O. Amidi, M. Bode, D. Bradley, A. Diaz-Calderon, M. Hoppold, H. Herman, R. Mandelbaum, T. Pilarki, P. Rander, S. Thayer, N. Vallidi, R. Warner: Toward reliable off road autonomous vehicles operating in challenging environments, *Int. J. Robot. Res.* **25**(5–6), 449–483 (2006)
- 36.14 I.S. Kweon, T. Kanade: High-resolution terrain map from multiple sensor data, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 278–292 (1992)
- 36.15 M. Montemerlo, S. Thrun: A multi-resolution pyramid for outdoor robot terrain perception, *Proc. AAAI Nat. Conf. Artif. Intell.* (San Jose 2004)
- 36.16 R. Triebel, P. Pfaff, W. Burgard: Multi-level surface maps for outdoor terrain mapping and loop closing, *IEEE/RSJ Int. Conf. Intell. Robot. Syst.* (2006)
- 36.17 C. Wellington, A. Courville, A. Stentz: A generative model of terrain for autonomous navigation in vegetation, *Int. J. Robot. Res.* **25**(12), 1287–1304 (2006)
- 36.18 P. Pfaff, R. Triebel, W. Burgard: An efficient extension to elevation maps for outdoor terrain mapping and loop closing, *Int. J. Robot. Res.* **26**(2), 217–230 (2007)
- 36.19 N. Fairfield, G. Kantor, D. Wettergreen: Real-time SLAM with octree evidence grids for exploration in underwater tunnels, *J. Field Robot.* **24**(1), 3–21 (2007)
- 36.20 A. Foessel: Scene Modeling from Motion-Free Radar Sensing. Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 2002)
- 36.21 J.-F. Lalonde, N. Vandapel, M. Hebert: Data structure for efficient processing in 3-D, *Proc. Robot. Sci. Syst. I* (2005) p. 48
- 36.22 J. Leal: Stochastic Environment Representation. Ph.D. Thesis (The University of Sydney, Sydney 2003)
- 36.23 P. Heckbert, M. Garland: Optimal triangulation and quadric-based surface simplification, *J. Comput. Geom. Theory Appl.* **14**(1–3), 49–65 (1999)
- 36.24 A. Akbarzadeh: Towards urban 3d reconstruction from video, *Int. Symp. 3D Data Proc. Visualization Transmission* (2006)
- 36.25 C. Frueh, S. Jain, A. Zakhor: Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images, *Int. J. Comput. Vis.* **61**(2), 159–184 (2005)
- 36.26 I. Stamos, P. Allen: Geometry and texture recovery of scenes of large scales, *Comput. Vis. Image Underst.* **88**, 94–118 (2002)
- 36.27 D. Gennery: Traversability analysis and path planning for a planetary rover, *Auton. Robot.* **6**, 131–146 (1999)
- 36.28 B. Sofman, E. Lin, J. Bagnell, J. Cole, N. Vandapel, A. Stentz: Improving robot navigation through self-supervised online learning, *J. Field Robot.* **23**(12), 1059–1075 (2006)
- 36.29 D. Ferguson, A. Stentz: The delayed D* algorithm for efficient path replanning, *Proc. IEEE Int. Conf. Robot. Autom.* (2005)
- 36.30 D. Ferguson, A. Stentz: Field D*: An interpolation-based path planner and replanner, *Proc. Int. Symp. Robot. Res. (ISRR)* (2005)
- 36.31 M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, S. Thrun: Anytime dynamic a*: An anytime, replanning algorithm, *Proc. Int. Conf. Autom. Planning Scheduling (ICAPS)* (2005)
- 36.32 D. Stavens, S. Thrun: A self-supervised terrain roughness estimator for off-road autonomous driving, *Uncertainty Artif. Intell.* (Boston 2006)
- 36.33 A. Angelova, L. Matthies, D. Helmick, P. Perona: Slip prediction using visual information, *Proc. Robot. Sci. Syst.* (Philadelphia 2006)
- 36.34 D. Kim, J. Sun, S. Oh, J. Rehg, A. Bobick: Traversability classification using unsupervised on-line visual learning for outdoor robot navigation, *IEEE Int. Conf. Robot. Autom.* (2006)
- 36.35 S. Thrun, M. Montemerlo, A. Aron: Probabilistic terrain analysis for high-speed desert driving, *Robotics Science and System Conference* (2005)
- 36.36 R. Murrieta-Cid, C. Parra, M. Devy: Visual navigation in natural environments: from range and color data to a landmark-based model, *Auton. Robot.* **13**(2), 143–168 (2002)
- 36.37 D. Asmar, J. Zelek, S. Abdallah: Tree trunks as landmarks for outdoor vision SLAM, *Proc. Conf. Comp. Vision Pattern Recognition Workshop* (2006)
- 36.38 I. Posner, D. Schroeter, P. Newman: Using scene similarity for place labelling, *Int. Symp. Exp. Robot.* (2006)
- 36.39 A. Torralba, K.P. Murphy, W.T. Freeman, M.A. Rubin: Context-based vision system for place and object recognition, *IEEE Int. Conf. Comput. Vis. (ICCV)* (2003)

- 36.40 D. Bradley, S. Thayer, A. Stentz, P. Rander: Vegetation detection for mobile robot navigation, Tech. Rep. **CMU-RI-TR-04-12**, Robotics Institute (Carnegie Mellon University, Pittsburgh 2004)
- 36.41 S. Kumar, J. Guivant, H. Durrant-Whyte: Informative representations of unstructured environments, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2004)
- 36.42 S. Kumar, F. Ramos, B. Douillard, M. Ridley, H. Durrant-Whyte: A novel visual perception framework, Proc. 9th Int. Conf. Contr. Autom. Robot. Vision (2006)
- 36.43 F. Ramos, S. Kumar, B. Upcroft, H. Durrant-Whyte: Representing natural objects in unstructured environments, Neural Inf. Proc. Syst. (NIPS) (2005)
- 36.44 C. Pantofaru, R. Unnikrishnan, M. Hebert: Toward generating labeled maps from color and range data for robot navigation, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003)
- 36.45 J.F. Lalonde, N. Vandapel, D. Huber, M. Hebert: Natural terrain classification using three-dimensional ladar data for ground robot mobility, J. Field Robot. **23**(10), 839–861 (2006)
- 36.46 M. Devy, R. Chatila, P. Fillatreau, S. Lacroix, F. Nashashibi: On autonomous navigation in a natural environment, Robot. Auton. Syst. **16**(1), 5–16 (1995)
- 36.47 R. Manduchi, A. Castano, A. Talukder, L. Matthies: Obstacle detection and terrain classification for autonomous off-road navigation, Auton. Robot. **18**(1), 81–102 (2005)
- 36.48 D. Huber, M. Hebert: 3d modeling using a statistical sensor model and stochastic search, Proc. IEEE Conf. Comput. Vision Pattern Recognition (CVPR) (2003) pp. 858–865
- 36.49 S. Balakirsky, A. Lacaze: World modeling and behavior generation for autonomous ground vehicles, IEEE Int. Conf. Robot. Autom. (2000)
- 36.50 A. Lacaze, K. Murphy, M. Delgiorno: Autonomous mobility for the demo III experimental unmanned vehicles, Proc. AUVSI (2002)
- 36.51 P. Bellutta, R. Manduchi, L. Matthies, K. Owens, A. Rankin: Terrain perception for demo III, Proc. Intell. Vehicles Symp. (2000)
- 36.52 J.F. Lalonde, R. Unnikrishnan, N. Vandapel, M. Hebert: Scale selection for classification of point-sampled 3-d surfaces, 5th Int. Conf. 3-D Digital Imaging Modeling (3DIM 2005) (2005)
- 36.53 J. Macedo, R. Manduchi, L. Matthies: Ladar-based discrimination of grass from obstacles for autonomous navigation, Proc. 7th Int. Symp. Exp. Robot. (ISER) (2000)
- 36.54 D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, A. Ng: Discriminative learning of Markov random fields for segmentation of 3-d scan data, Proc. Conf. Comp. Vision Pattern Recognition (2005)
- 36.55 R. Triebel, K. Kersting, W. Burgard: Robust 3d scan point classification using associative Markov networks, IEEE Int. Conf. Robot. Autom. (2006)
- 36.56 H. Chen, P. Meer, D. Tyler: Robust regression for data with multiple structures, IEEE Int. Conf. Comput. Vision Pattern Recognition (2001)
- 36.57 R. Unnikrishnan, M. Hebert: Robust extraction of multiple structures from non-uniformly sampled data, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003)
- 36.58 D. Wolf, G. Sukhatme, D. Fox, W. Burgard: Autonomous terrain mapping and classification using hidden Markov models, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2005)
- 36.59 C. Olson, L. Matthies, J. Wright, R. Li, K. Di: Visual terrain mapping for Mars exploration, Comput. Vis. Understand. **105**, 73–85 (2007)
- 36.60 J. Nieto, J. Guivant, E. Nebot: The hybrid metric maps (hymms): a novel map representation for denseSLAM, IEEE Int. Conf. Robot. Autom. (2004)
- 36.61 F. Ramos, J. Nieto, H. Durrant-Whyte: Recognising and modelling landmarks to close loops in outdoor SLAM, IEEE Int. Conf. Robot. Autom. (2007)
- 36.62 D. Hähnel, D. Schulz, W. Burgard: Mobile robot mapping in populated environments, Adv. Robot. **17**(7), 579–598 (2003)
- 36.63 C.-C. Wang, C. Thorpe, S. Thrun: Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2003)
- 36.64 P. Biber, T. Duckett: Dynamic maps for long-term operation of mobile service robots, Proc. Robot. Sci. Syst. (RSS) (2005)
- 36.65 C. Stachniss, W. Burgard: Mobile robot mapping and localization in non-static environments, Proc. Nat. Conf. Artif. Intell. (Pittsburgh 2005)
- 36.66 D. Hähnel, R. Triebel, W. Burgard, S. Thrun: Map building with mobile robots in dynamic environments, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2003)
- 36.67 R. Siegwart, I. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT-Press, Cambridge 2001)
- 36.68 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
- 36.69 H. Samet: *Foundations of Multidimensional and Metric Data Structures* (Elsevier, Amsterdam 2006)

37. Simultaneous Localization and Mapping

Sebastian Thrun, John J. Leonard

This chapter provides a comprehensive introduction in to the *simultaneous localization and mapping problem*, better known in its abbreviated form as **SLAM**. **SLAM** addresses the problem of a robot navigating an unknown environment. While navigating the environment, the robot seeks to acquire a map thereof, and at the same time it wishes to localize itself using its map. The use of **SLAM** problems can be motivated in two different ways: one might be interested in detailed environment models, or one might seek to maintain an accurate sense of a mobile robot's location. **SLAM** serves both of these purposes.

We review three major paradigms of algorithms from which a huge number of recently published methods are derived. First comes the traditional approach, which relies on the extended Kalman filter (**EKF**) for representing the robot's best estimate. The second paradigm draws its intuition from the fact that the **SLAM** problem can be viewed as a sparse graph of constraints, and it applies nonlinear optimization for recovering the map and the robot's locations. Finally, we survey the particle filter paradigm, which applies nonparametric

37.1	Overview	871
37.2	SLAM: Problem Definition	872
37.2.1	Mathematical Basis	872
37.2.2	Example: SLAM in Landmark Worlds	873
37.2.3	Taxonomy of the SLAM Problem.....	873
37.3	The Three Main SLAM Paradigms	875
37.3.1	Extended Kalman Filters	875
37.3.2	Graph-Based Optimization Techniques.....	877
37.3.3	Particle Methods	881
37.3.4	Relation of Paradigms	883
37.4	Conclusion and Future Challenges	885
37.5	Suggestions for Further Reading	886
	References	886

density estimation and efficient factorization methods to the **SLAM** problem. This chapter discusses extensions of these basic methods. It elucidates variants of the **SLAM** problem and proposes a taxonomy for the field. Relevant research is referenced extensively, and open research problems are discussed.

37.1 Overview

This chapter provides a comprehensive introduction into one of the key enabling technologies of mobile robot navigation: *simultaneous localization and mapping* (**SLAM**). **SLAM** addresses the problem of acquiring a spatial map of a mobile robot environment while simultaneously localizing the robot relative to this model. The **SLAM** problem is generally regarded as one of the most important problems in the pursuit of building truly autonomous mobile robots. Despite significant progress in this area, it still poses great challenges. At present, we have robust methods for

mapping environments that are static, structured, and of limited size. Mapping unstructured, dynamic, or large-scale environments remains largely an open research problem.

The historical roots of **SLAM** can be traced back to *Gauss* [37.1], who is largely credited with inventing the least-squares method, for calculating planetary orbits. In the 20th century, a number of fields outside robotics have studied the making of environment models from a moving sensor platform, most notably in *photogrammetry* [37.2] and *computer vision* [37.3,4]. **SLAM** builds

on this work, often extending the basic paradigms into more scalable algorithms.

This chapter begins with a definition of the **SLAM** problem, which shall include a brief taxonomy of different versions of the problem. The centerpiece of this chapter is a layman's introduction into the three major paradigms in this field, and the various extensions that exist. As the reader will quickly recognize, there is no single best solution to the **SLAM** problem. The method

chosen by the practitioner will depend on a number of factors, such as the desired map resolution, the update time, the nature of the features in the map, and so on. Nevertheless, the three methods discussed in this chapter cover the major paradigms in this field. For an in-depth study of **SLAM** algorithms, we refer the reader to a recent textbook on probabilistic robotics, which dedicates a number of chapters to the topic of **SLAM** [37.5]. Also see [37.6, 7] for a recent in-depth tutorial for **SLAM**.

37.2 SLAM: Problem Definition

37.2.1 Mathematical Basis

The **SLAM** problem is defined as follows. A mobile robot roams an unknown environment, starting at a location with known coordinates. Its motion is uncertain, making it gradually more difficult to determine its global coordinates. As it roams, the robot can sense its environment. The **SLAM** problem is the problem of building a map of the environment while simultaneously determining the robot's position relative to this map.

Formally, **SLAM** is best described in probabilistic terminology. Let us denote time by t , and the robot location by x_t . For mobile robots on a flat ground, x_t is usually a three-dimensional vector, consisting of its two-dimensional coordinate in the plane plus a single rotational value for its orientation. The sequence of locations, or *path*, is then given as

$$X_T = \{x_0, x_1, x_2, \dots, x_T\}. \quad (37.1)$$

Here T is some terminal time (T might be ∞). The initial location x_0 is known; other positions cannot be sensed.

Odometry provides relative information between two consecutive locations. Let u_t denote the odometry that characterized the motion between time $t-1$ and time t ; such data might be obtained from the robot's wheel encoders or from the controls given to those motors. Then the sequence

$$U_T = \{u_1, u_2, u_3 \dots x_T\} \quad (37.2)$$

characterizes the relative motion of the robot. For noise-free motion, U_T would be sufficient to recover the past X_T from the initial location x_0 . However, odometry measurements are noisy, and path-integration techniques inevitably diverge from the truth.

Finally, the robot senses objects in the environment. Let m denote the *true* map of the environment. The

environment may be comprised of landmarks, objects, surfaces, etc., and m describes their locations. The environment map m is typically assumed to be time invariant (i. e., static).

The robot measurements establish information between features in m and the robot location x_t . If we, without loss of generality, assume that the robot takes exactly one measurement at each point in time, the sequence of measurements is given as

$$Z_T = \{z_1, z_2, z_3, \dots, z_T\}. \quad (37.3)$$

Figure 37.1 illustrates the variables involved in the **SLAM** problem. It shows the sequence of locations and sensor measurements, and the causal relationships between these variables. Such a diagram is known as a *graphical model*. It is useful in understanding the dependencies in the **SLAM** problem.

The **SLAM** problem is now the problem of recovering a model of the world m and the sequence of robot locations X_T from the odometry and measure-

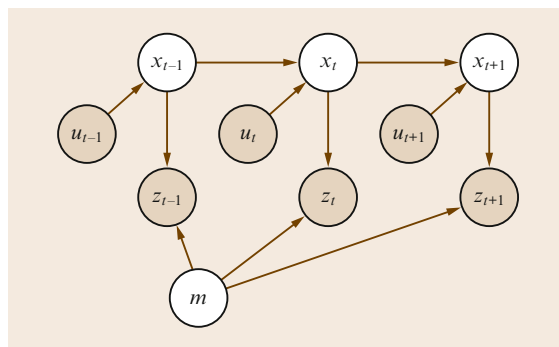


Fig. 37.1 Graphical model of the **SLAM** problem. Arcs indicate causal relationships, and shaded nodes are directly observable to the robot. In **SLAM**, the robot seeks to recover the unobservable variables

ment data. The literature distinguishes two main forms of the **SLAM** problem, which are both of equal practical importance. One is known as the *full SLAM problem*: it involves estimating the posterior over the entire robot path together with the map:

$$p(X_T, m | Z_T, U_T). \quad (37.4)$$

Written in this way, the full **SLAM** problem is the problem of calculating the joint posterior probability over X_T and m from the available data. Notice that the variables right of the conditioning bar are all directly observable to the robot, whereas those on the left are the ones that we want. As we shall see, algorithms for the offline **SLAM** problem are often batch, that is, they process all data at the same time.

The second, equally important **SLAM** problem is the *online SLAM problem*. This problem is defined via

$$p(x_t, m | Z_T, U_T). \quad (37.5)$$

Online **SLAM** seeks to recover the present robot location, instead of the entire path. Algorithms that address the online problem are usually incremental and can process one data item at a time. In the literature such algorithms are typically called *filters*.

To *solve* either **SLAM** problem, the robot needs to be endowed with two more models: a mathematical model that relates odometry measurements u_t to robot locations x_{t-1} and x_t and a model that relates measurements z_t to the environment m and the robot location x_t . These models corresponds to the arcs in Fig. 37.1.

In **SLAM**, it is common to think of those mathematical models as probability distributions: $p(x_t | x_{t-1}, u_t)$ characterizes the probability distribution of the location x_t assuming that a robot started at a known location x_{t-1} and measured odometry data u_t , and likewise $p(z_t | x_t, m)$ is the probability for measuring z_t if this measurement is taken at a known location x_t in a known environment m . Of course, in the **SLAM** problem we do *not* know the robot location, and neither do we know the environment. As we shall see, Bayes rule takes care of this, by transforming these mathematical relationships into a form where we can *recover* probability distributions over those latent variables from the measured data.

37.2.2 Example: SLAM in Landmark Worlds

One common setting of **SLAM** involves the assumption that the environment is populated by point landmarks. When building two-dimensional (2-D) maps, point landmarks may correspond to door posts and corners of

rooms, which, when projected into a 2-D map are characterized by a point coordinate. In a 2-D world, each point landmark is characterized by two coordinate values. Hence the world is a vector of size $2N$, where N is the number of point landmarks in the world.

In a commonly studied setting, the robot can sense three things: the relative range to nearby landmarks, their relative bearing, and the identity of these landmarks. The range and bearing may be noisy, but in the most simple case the identity of the sensed landmarks is known perfectly.

To model such a setup, one begins with defining the *exact*, noise-free measurement function. The measurement function h describes the workings of the sensors: it accepts as input a description of the environment m and a robot location x_t , and it computes the measurement:

$$h(x_t, m). \quad (37.6)$$

Computing h is straightforward in our simplified landmark setting; it is a simple exercise in trigonometry.

The probabilistic measurement model is derived from this measurement function by adding a noise term; it is a probability distribution that peaks at the noise-free value $h(x_t, m)$ but allows for measurement noise:

$$p(z_t | x_t, m) \sim \mathcal{N}(h(x_t, m), Q_t). \quad (37.7)$$

Here \mathcal{N} denotes the two-dimensional normal distribution, which is centered at $h(x_t, m)$. The 2-by-2 matrix Q_t is the noise covariance, indexed by time.

The motion model is derived from a kinematic model of robot motion. Given the location vector x_{t-1} and the motion u_t , textbook kinematics tells us how to calculate x_t . Let this function be denoted by g :

$$g(x_{t-1}, u_t). \quad (37.8)$$

The motion model is then defined by a normal distribution centered at $g(x_{t-1}, u_t)$ but subject to Gaussian noise:

$$p(x_t | x_{t-1}, u_t) = \mathcal{N}(g(x_{t-1}, u_t), R_t). \quad (37.9)$$

Here R_t is a covariance; it is of size 3-by-3, since the location is a three-dimensional vector.

With these definitions, we have all we need to develop a **SLAM** algorithm. While in the literature, point landmark problems with range-bearing sensing are by far the most widely studied, **SLAM** algorithms are not confined to landmark worlds. However, no matter what the map representation and the sensor modality, any **SLAM** algorithm needs a similarly crisp definition of

the features in m , the measurement model $p(z_t | x_t, m)$, and the motion model $p(x_t | x_{t-1}, u_t)$.

37.2.3 Taxonomy of the SLAM Problem

SLAM problems are distinguished along a number of different dimensions. Most important research papers identify the type of problems addressed by making the underlying assumptions explicit. We already encountered one such distinction: full versus online. Other common distinctions are as follows.

Volumetric Versus Feature-Based

In volumetric SLAM, the map is sampled at a resolution high enough to allow for photorealistic reconstruction of the environment. The map m in volumetric SLAM is usually quite high dimensional, with the result that the computation can be quite involved. Feature-based SLAM extracts sparse features from the sensor stream. The map is then only comprised of features. Our point-landmark example is an instance of feature-based SLAM. Feature-based SLAM techniques tend to be more efficient, but their results may be inferior to volumetric SLAM due to the fact that the extraction of features discards information in the sensor measurements.

Topological Versus Metric

Some mapping techniques recover only a qualitative description of the environment, which characterizes the relation of basic locations. Such methods are known as topological. A topological map might be defined over a set of distinct places and a set of qualitative relations between these places (e.g., place A is adjacent to place B). Metric SLAM methods provide metric information between the relation of such places. In recent years, topological methods have fallen out of fashion, despite ample evidence that humans often use topological information for navigation.

Known Versus Unknown Correspondence

The correspondence problem is the problem of relating the identity of sensed things to other sensed things. In the landmark example above, we assumed that the identity of landmarks is known. Some SLAM algorithms make such an assumption, while others do not. The algorithms that do not make this assumption provide special mechanisms for estimating the correspondence of measured features to previously observed landmarks in the map. The problem of estimating the correspondence is known as the *data association problem*, and is one of the most difficult problems in SLAM.

Static Versus Dynamic

Static SLAM algorithms assume that the environment does not change over time. Dynamic methods allow for changes in the environment. The vast majority of the literature on SLAM assumes static environments; dynamic effects are often treated just as measurement outliers. Methods that reason about motion in the environment are more involved, but they tend to be more robust in most applications.

Small Versus Large Uncertainty

SLAM problems are distinguished by the degree of location uncertainty that they can handle. The most simple SLAM algorithms allow only for small errors in the location estimate. They are good for situations in which a robot goes down a path that does not intersect itself, and then returns along the same path. In many environments it is possible to reach the same location from multiple directions. Here the robot may accrue a large amount of uncertainty. This problem is known as the *loop-closing problem*. When closing a loop, the uncertainty may be large. The ability to close loops is a key characteristic of modern-day SLAM algorithms. The uncertainty can be reduced if the robot can sense information about its position in some absolute coordinate frame, e.g., through the use of a satellite-based global positioning system (GPS) receiver.

Active Versus Passive

In passive SLAM algorithms, some other entity controls the robot, and the SLAM algorithm is purely observing. The vast majority of algorithms are of this type; they give the robot designer the freedom to implement arbitrary motion controllers, and pursue arbitrary motion objectives. In active SLAM, the robot actively explores its environment in the pursuit of an accurate map. Active SLAM methods tend to yield more accurate maps in less time, but they constrain the robot motion. There exist hybrid techniques in which the SLAM algorithm controls only the pointing direction of the robot's sensors, but not the motion direction.

Single-Robot Versus Multirobot SLAM

Most SLAM problems are defined for a single-robot platform, although recently the problem of multirobot exploration has gained in popularity. Multirobot SLAM problems come in many flavors. In some, robots are able to observe each other, while in others robots are told their relative initial locations. Multirobot SLAM problems are also distinguished by the type of communication allowed between the different robots. In some, the robots can

communicate with no latency and infinite bandwidth. More realistic are setups in which only nearby robots can communicate, and the communication is subject to latency and bandwidth limitations.

As this taxonomy suggests, there exists a flurry of SLAM algorithms. Most modern-day conferences

dedicate multiple sessions to SLAM. This chapter focuses on the very basic SLAM setup. In particular it assumes a static environment with a single robot. Extensions are discussed towards the end of this chapter, in which the relevant literature is also discussed.

37.3 The Three Main SLAM Paradigms

This section reviews three basic SLAM paradigms, from which most others are derived. The first, known as extended Kalman filter (EKF) SLAM, is historically the earliest but has recently become slightly unpopular due to its limiting computational properties. The second, which is based on graphical representations, successfully applies sparse nonlinear optimization methods to the SLAM problem, and has become the main paradigm for solving the full SLAM problem. The third and final method uses nonparametric statistical filtering techniques known as particle filters. It is a popular method for online SLAM, and provides a fresh new solution to the data association problem in SLAM.

37.3.1 Extended Kalman Filters

Historically, the EKF formulation of SLAM is the earliest, and perhaps the most influential. EKF SLAM was introduced in [37.8–10] and [37.11, 12], which were the first papers to propose the use of a single state vector to estimate the locations of the robot and a set of features in the environment, with an associated error covariance matrix representing the uncertainty in these estimates, including the correlations between the vehicle and feature state estimates. As the robot moves through its environment taking measurements, the system state vector and covariance matrix are updated using the extended Kalman filter [37.13–15]. As new features are observed, new states are added to the system state vector; the size of the system covariance matrix grows quadratically.

This approach assumes a metrical, feature-based environmental representation, in which objects can be effectively represented as points in an appropriate parameter space. The position of the robot and the locations of features form a network of uncertain spatial relationships. The development of appropriate representations is a critical issue in SLAM, and intimately related to the topics of sensing and world modeling discussed in Chap. 36 and in Part C in this handbook.

The EKF algorithm represents the robot estimate by a multivariate Gaussian:

$$p(x_t, m \mid Z_T, U_T) \sim \mathcal{N}(\mu_t, \Sigma_t). \quad (37.10)$$

The high-dimensional vector μ_t contains the robot's best estimate of its own location and the location of the features in the environment. In our point-landmark example, the dimension of μ_t would be $3 + 2N$, since we need three variables to represent the robot location and $2N$ variables for the N landmarks in the map.

The matrix Σ_t is the covariance of the robot's assessment of its expected error in the guess μ_t . As a quadratic matrix, Σ_t is of size $(3 + 2N) \times (3 + 2N)$. In SLAM, this matrix is usually distinctly non-sparse. The off-diagonal elements capture the correlations in the estimates of different variables. Nonzero correlations come along because the robot's location is uncertain, and as a result the locations of the landmarks in the maps are uncertain. The importance of maintaining those off-diagonal elements is one of the key properties of EKF SLAM [37.16].

The EKF SLAM algorithm is easily derived for our point-landmark example. Suppose, for a moment that the motion function g and the measurement function h were linear in their arguments. Then the vanilla Kalman filter, as described in any textbook on Kalman filtering, would be applicable. EKF SLAM linearizes the functions g and h using Taylor-series expansion – again, this is standard textbook material. Thus, in its most basic form (and in the absence of any data association problems), EKF SLAM is nothing but the application of the vanilla EKF to the online SLAM problem.

Figure 37.2 illustrates the EKF SLAM algorithm for an artificial example. The robot navigates from a start pose that serves as the origin of its coordinate system. As it moves, its own pose uncertainty increases, as indicated by uncertainty ellipses of growing diameter. It also senses nearby landmarks and maps them with an uncertainty that combines the fixed measurement uncertainty with the increasing pose uncertainty. As a result, the uncertainty in the landmark locations grows over time. The

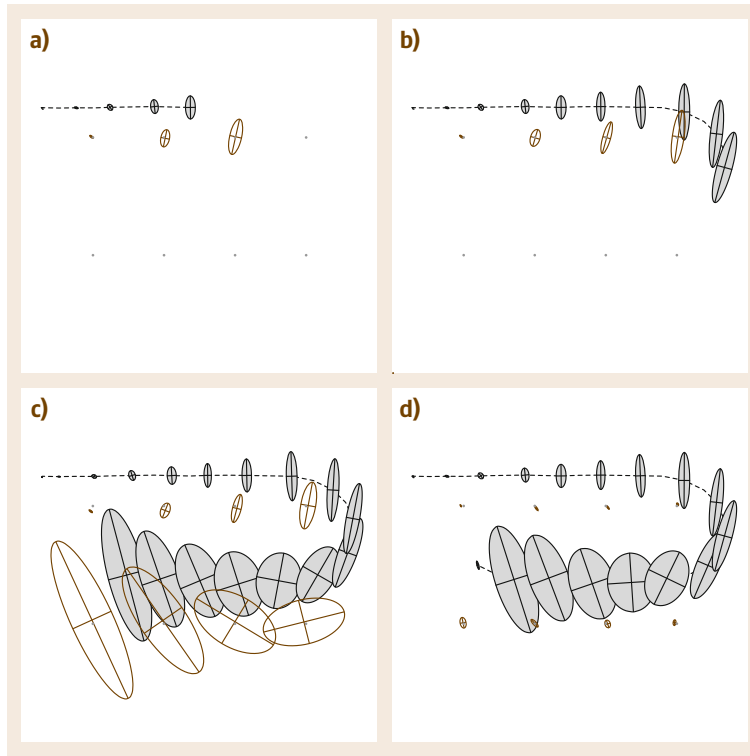


Fig. 37.2a–d EKF applied to the on-line SLAM problem. The robot's path is a dotted line, and its estimates of its own position are shaded ellipses. Eight distinguishable landmarks of unknown location are shown as small dots, and their location estimates are shown as white ellipses. In (a–c) the robot's positional uncertainty is increasing, as is its uncertainty about the landmarks it encounters. In (d) the robot senses the first landmark again, and the uncertainty of all landmarks decreases, as does the uncertainty of its current pose. (Image courtesy of Michael Montemerlo, Stanford University)

interesting transition is illustrated in Fig. 37.2d: Here the robot observes the landmark it saw in the very beginning of mapping, and whose location is relatively well known. Through this observation, the robot's pose error is reduced, as indicated in Fig. 37.2d – notice the very small error ellipse for the final robot pose. This observation also reduces the uncertainty for other landmarks in the map. This phenomenon arises from a correlation that is expressed in the covariance matrix of the Gaussian posterior. Since most of the uncertainty in earlier landmark estimates is caused by the robot pose, and since this very uncertainty persists over time, the location estimates of those landmarks are correlated. When gaining information on the robot's pose, this information spreads to previously observed landmarks. This effect is probably the most important characteristic of the SLAM posterior [37.16]. Information that helps localize the robot is propagated through the map, and as a result improves the localization of other landmarks in the map.

EKF SLAM also addresses the *data association problem*. If the identity of observed features is unknown, the basic EKF idea becomes inapplicable. The solution here is to reason about the most likely data association

when a landmark is observed. This is usually done based on proximity: *which of the landmarks in the map most likely corresponds to the landmark just observed?* The proximity calculation considers the measurement noise and the actual uncertainty in the poster estimate, and the metric used in this calculation is known as a Mahalanobis distance, which is a weighted quadratic distance. To minimize the chances of false data associations, many implementations use visible features to distinguish individual landmarks and associate groups of landmarks observed simultaneously [37.17, 18]. Modern-day implementations also maintain a *provisional landmark list* and only add landmarks to the internal map when they have been observed sufficiently frequently [37.19–22]. With an appropriate landmark definition and careful implementation of the data association step, EKF SLAM becomes a powerful method for feature-based SLAM.

EKF SLAM has been applied successfully to a large range of navigation problems, involving airborne, underwater, indoor, and various other vehicles. Figure 37.3a shows an example result obtained with a state-of-the-art implementation of EKF SLAM. Shown there is an underwater map obtained with the underwater robot Oberon, developed at the University of Sydney, Aus-

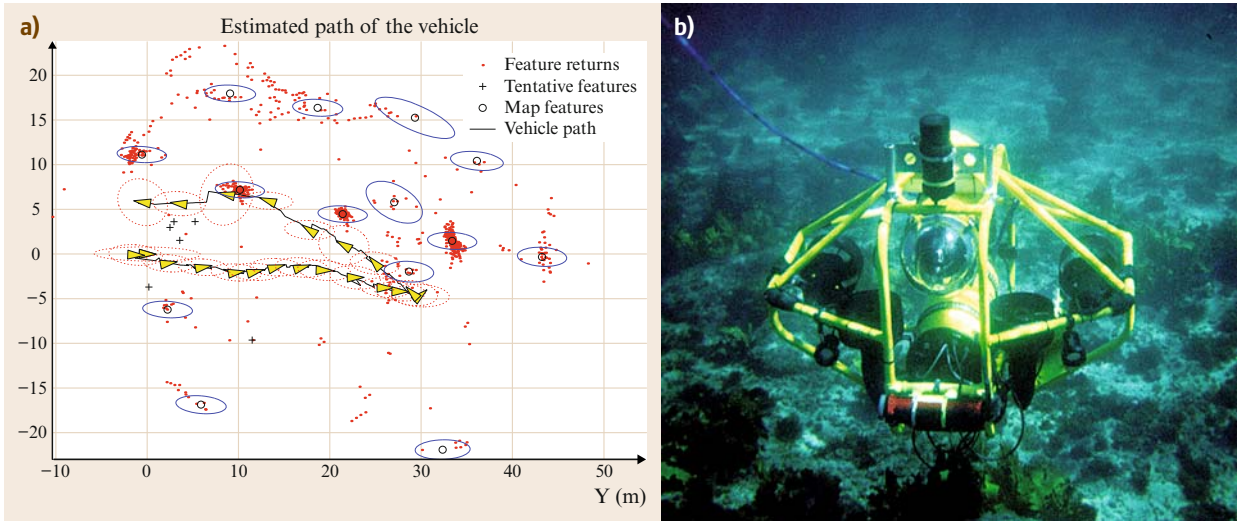


Fig. 37.3 (a) Example of Kalman filter estimation of the map and the vehicle pose. (b) The underwater vehicle Oberon, developed at the University of Sydney. (Images courtesy of Stefan Williams and Hugh Durrant-Whyte, Australian Centre for Field Robotics)

tralia, and shown in Fig. 37.3b. This vehicle is equipped with a pencil-beam, mechanically scanned sonar that can produce high-resolution range and bearing measurements of objects up to 50 m away. To facilitate the mapping problem, researchers have deposited long, small vertical objects in the water, which can be extracted from the sonar scans with relative ease. In this specific experiment, there is a row of such objects, spaced approximately 10 m apart. In addition, a more distant cliff offers additional point features that can be detected using the scanning sonar.

The map shown in Fig. 37.3a depicts the robot path, marked by the triangles connected by a line. Around each triangle one can see an ellipse, which corresponds to the covariance matrix of the Kalman filter estimate projected into the robot location. The ellipse shows the variance; the larger it is, the less certain the robot is about its current pose. Various small dots in Fig. 37.3a show landmark sightings, obtained by searching the sonar scan for small and highly reflective objects. The majority of these sightings is rejected using statistical outlier rejection techniques [37.20]. However, some are believed to correspond to a landmark and are added to the map. At the end of the run, the robot has classified 14 such objects as landmarks, each of which is plotted with the projected uncertainty ellipse in Fig. 37.3a. These landmarks include the artificial landmarks put out by the researchers, but they also include various other terrain features in the vicinity of the robot. The residual pose uncertainty is small.

The basic formulation of **EKF SLAM** assumes that the location of features in the map is fully observable from a single position of the robot. The method has been extended to situations with partial observability, with range-only [37.23] or angle-only [37.24, 25] measurements. The technique has also been utilized using a featureless representation, in which the state consists of current and past robot poses, and measurements take the form of constraints between the poses (derived, for example, from laser scan matching or from camera measurements) [37.26, 27].

A key concern of the **EKF** approach to **SLAM** lies in the quadratic nature of the covariance matrix. A number of researchers have proposed extensions to the **EKF SLAM** algorithms that gain remarkable scalability by decomposing the map into submaps, for which covariances are maintained separately. Relevant literature can be found in [37.19, 28–32]. Other researchers have developed hybrid **SLAM** techniques, which combine **EKF**-style **SLAM** techniques with volumetric map representation; see [37.33–36]. Finally, researchers have developed data association techniques for **SLAM** [37.37–39] based on advanced statistical techniques such as Dempster’s **EM** algorithm [37.40].

37.3.2 Graph-Based Optimization Techniques

A second family of algorithms solves the **SLAM** problem through nonlinear sparse optimization. They draw

their intuition from a graphical representation of the **SLAM** problem. Graph-based techniques were first mentioned in [37.8, 41], but a seminal paper [37.42] provided a first working solution. The representation in this section is closely related to a series of recent papers [37.43–52]. We note that most contemporary techniques are offline and address the full **SLAM** problem, although some online versions exist and will be discussed below.

The basic intuition of graph-based **SLAM** is a follows. Landmarks and robot locations can be thought of as nodes in a graph. Every consecutive pair of locations x_{t-1}, x_t is tied together by an arc that represents the information conveyed by the odometry reading u_t . Other arcs exist between locations x_t and landmarks m_i , assuming that at time t the robot sensed landmark i . Arcs in this graph are soft constraints. Relaxing these constraints yields the robot's best estimate for the map and the full path.

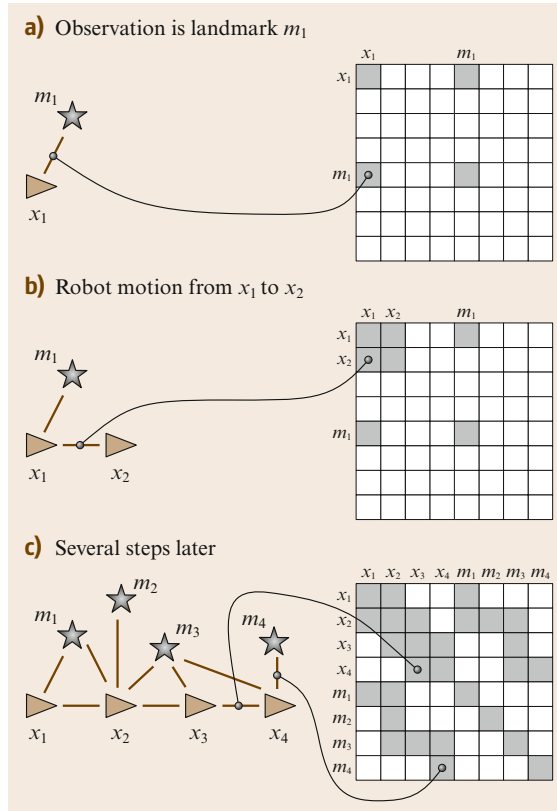


Fig. 37.4a–c Illustration of the graph construction. The *left* diagram shows the graph, the *right* the constraints in matrix form

The construction of the graph is illustrated in Fig. 37.4. Suppose that at time $t = 1$ the robot senses landmark m_1 . This adds an arc in the (yet highly incomplete) graph between x_1 and m_1 . When caching the edges in a matrix format (which happens to correspond to a quadratic equation defining the resulting constraints), a value is added to the elements between x_1 and m_1 , as shown on the right-hand side of Fig. 37.4a.

Now suppose the robot moves. The odometry reading u_2 leads to an arc between nodes x_1 and x_2 , as shown in Fig. 37.4b. Consecutive application of these two basic steps leads to an graph of increasing size, as illustrated in Fig. 37.4c. Nevertheless this graph is *sparse*, in that each node is only connected to a small number of other nodes. The number of constraints in the graph is (at worst) *linear* in the time elapsed and in the number of nodes in the graph.

If we think of the graph as a spring–mass model [37.50], computing the **SLAM** solution is equivalent to computing the state of minimal energy of this model. To see this, note that the graph corresponds to the log-posterior of the full **SLAM** problem (cf. (37.4)):

$$\log p(X_T, m \mid Z_T, U_T). \quad (37.11)$$

Without derivation, we state that this logarithm is of the form

$$\begin{aligned} \log p(X_T, m \mid Z_T, U_T) = \text{const} \\ + \sum_t \log p(x_t \mid x_{t-1}, u_t) \\ + \sum_t \log p(z_t \mid x_t, m). \end{aligned} \quad (37.12)$$

Each constraint of the form $\log p(x_t \mid x_{t-1}, u_t)$ is the result of exactly one robot motion event, and it corresponds to an arc in the graph. Likewise, each constraint of the form $\log p(z_t \mid x_t, m)$ is the result of one sensor measurement, to which we can also find a corresponding arc in the graph. The **SLAM** problem is then simply to find the mode of this equation.

$$X_T^*, m^* = \operatorname{argmax}_{X_T, m} \log p(X_T, m \mid Z_T, U_T). \quad (37.13)$$

Without derivation, we note that, under the Gaussian noise assumption that was made in the point-landmark example, this expression resolves to the following

quadratic form:

$$\begin{aligned} \log p(X_T, m \mid Z_T, U_T) \\ = \text{const} \\ + \sum_t [x_t - g(x_{t-1}, u_t)]^T R_t^{-1} [x_t - g(x_{t-1}, u_t)] \\ + \sum_t [z_t - h(x_t, m)]^T Q_t^{-1} [z_t - h(x_t, m)]. \end{aligned} \quad (37.14)$$

This is a sparse function. A number of efficient optimization techniques can be applied. Common choices include gradient descent, conjugate gradient, and others. Most SLAM implementations rely on some sort of iterative linearizing the functions g and h , in which case the objective in (37.14) becomes quadratic in all of its variables.

The graphical paradigm is easily extended to handle the data association problems. This is because (37.14) is easily extended to integrate additional knowledge on data association. Suppose some oracle informed us that landmarks m_i and m_j in the map corresponded to one and the same physical landmark in the world. Then we can either remove m_j from the graph and attach all adjacent arcs to m_i , or we can add a *soft correspondence constraint* [37.53] of the form

$$[m_j - m_i]^T \Gamma [m_j - m_i]. \quad (37.15)$$

Here Γ is a 2-by-2 diagonal matrix whose coefficients determine the penalty for *not* assigning identical locations to two landmarks (hence we want Γ to be large). Since graphical methods are usually used for the full SLAM problem, the optimization can be interleaved

with the search for the optimal data association. State-of-the-art implementations rely on RANSAC [37.54] or branch-and-bound methods [37.55, 56].

Graphical SLAM methods have the advantage that they scale to much higher-dimensional maps than EKF SLAM. The key limiting factor in EKF SLAM is the covariance matrix, which takes space (and update time) quadratic in the size of the map. No such constraint exists in graphical methods. The update time of the graph is constant, and the amount of memory required is linear (under some mild assumptions). Performing the optimization can be expensive, however. Technically, finding the optimal data association is suspected to be an NP-hard problem, although in practice the number of plausible assignments is usually small. The continuous optimization of the log-likelihood function in (37.14) depends among other things on the number and size of loops in the map.

Figure 37.5 shows the result of a state-of-the-art SLAM algorithm based on analyzing the constraint graph and a nested search of the best data association. The data for this map was acquired by CMU's Groundhog robot [37.57], built to explore and map abandoned underground mines. Groundhog is equipped with a laser range finder which measures the range to obstacles along a horizontal slice of the world. The specific map shown here covers an area of 250 m \times 150 m. The form of the map is known as a *occupancy grid map*, which is due to Elfes and Moravec [37.58, 59]. Occupancy grid maps use Bayesian reasoning to estimate the posterior probability that a cell is free, thereby accommodating noise in range finders.

As a baseline for comparison, Fig. 37.5a shows a map constructed in a much simpler way: here scans

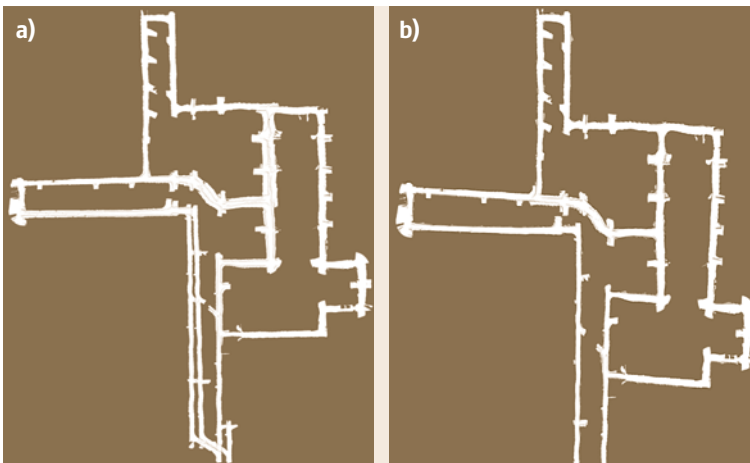


Fig. 37.5a,b An occupancy grid map of an abandoned mine: (a) estimates the data association incrementally, and only in reference to the most recent sensor measurement. The right map (b) is the result of a global data association search and a graphical optimization (Images courtesy of Dirk Hähnel, University of Freiburg)

are localized relative to slightly older scans and, once localized, are added to the map under the assumption that the estimated location is correct. Such a technique is called *scan matching* [37.42]. Scan matching is a *SLAM* method, but it can only accommodate very small amounts of location uncertainties. The failure to close loops is obvious from Fig. 37.5a. In fact, pairwise scan matching can be thought of as a version of a graphical *SLAM* algorithm, but correspondence is only established (and constraints inserted in the graph) between immediately consecutive scans.

To map this data into a graph of manageable size, the algorithm decomposes the map into small local submaps, one for each 5 m of robot travel. Within these 5 m, the maps are sufficiently accurate, as general drift is small and hence scan matching performs essentially flawlessly. Each submap coordinates become a pose node in the GraphSLAM. Adjacent submaps are linked through the relative motion constraints between them. The resulting structure is shown in Fig. 37.5b.

On this graph, we can now perform a branch-and-bound recursive search for correspondences. For finding

good submaps that might correspond, this algorithm uses a correlation analysis for two overlaying maps. Once two suitable maps are found, a soft constraint of the type stated in (37.15) is added to the graph, followed by an optimization step of the resulting set of constraints. Figure 37.6 illustrates the process of data association: each circle corresponds to a new constraint that would be found in the search. The figure illustrates the iterative nature of the search: certain correspondences are only discovered when others have been propagated, and others are dissolved in the process of the search. The final model is stable, in that additional search for new data association induces no further changes. The resulting grid map is shown in Fig. 37.5b.

Other graph-based techniques for *SLAM* have produced similar results. Figure 37.7 shows a map of the same data set generated by [37.26], using an algorithm called *Atlas*. This algorithm decomposes maps into submaps whose relation is maintained through information-theoretic relative link.

We note that the graph-based paradigm is very closely linked to information theory, in that the soft con-

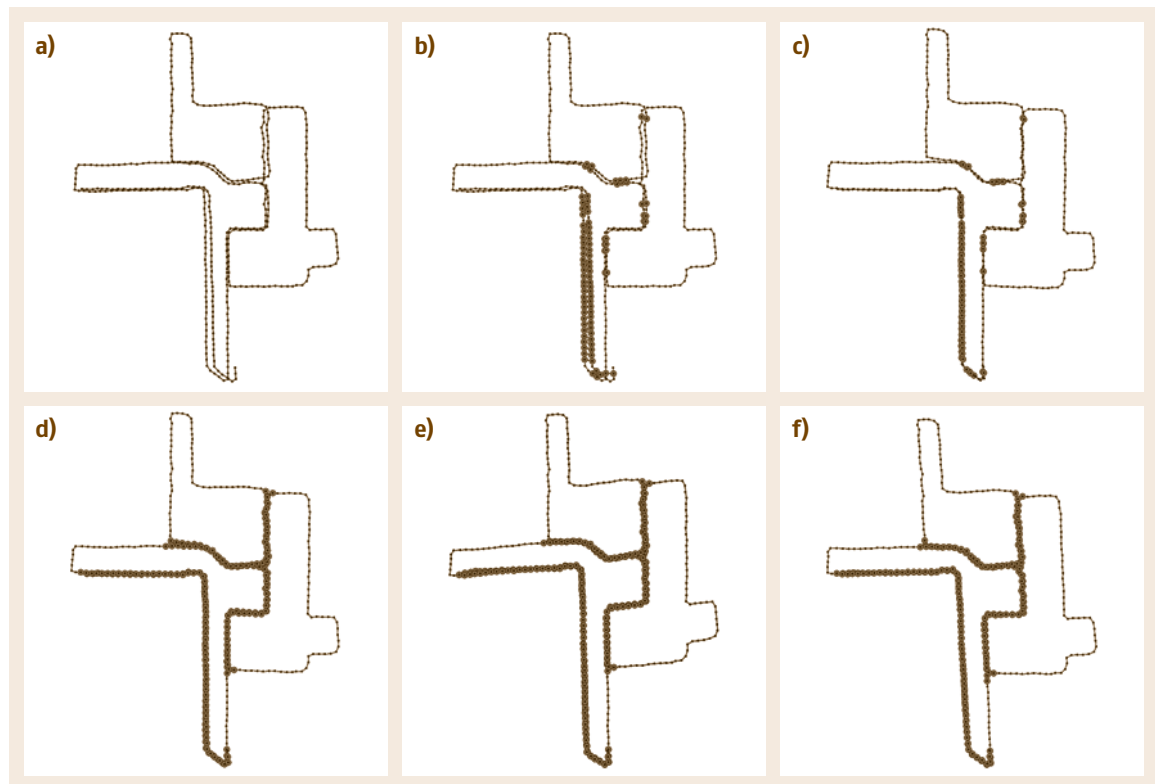


Fig. 37.6a–f Data association search. See text

straints constitute the information the robot has on the world (in an information-theoretic sense [37.60]). Most methods in the field are inherently offline, that is, they optimize for the entire robot path. If the robot path is long, the optimization may become cumbersome. This is one of the disadvantages of the graph-based paradigm. There exists a number of crossovers that manipulate the graph online so as to factor our past robot location variables. The resulting algorithms are filters [37.26, 61–63] and they tend to be intimately related to information filter methods [37.16, 63–67]. Many of the original attempts to decompose EKF SLAM representations into smaller submaps to scale up are based on motivations that are not dissimilar to the graphical approach; see [37.29, 30, 68].

As this chapter is being written, graphical and optimization-based SLAM algorithm are the subject of intense research. Recent results show that the paradigm scales to maps with 10^8 features [37.26, 43–49, 51, 52, 57]. Arguably, the graph-based paradigm has generated some the largest SLAM maps ever built, but usually in an offline fashion.

37.3.3 Particle Methods

The third principal SLAM paradigm is based on particle filters. Particle filters can be traced back to [37.69], but they have become popular only in recent years. Particle filter represent a posterior through a set of *particles*. For the novice in SLAM, each particle is best thought as a concrete guess as to what the true value of the state may be. By collecting many such guesses into a set of guesses, or set of particles, the particle filters captures a representative sample from the posterior distribution. The particle filter has been shown under mild conditions to approach the true posterior as the particle set size goes to infinity. It is also a nonparametric representation that represents multimodal distributions with ease. In recent years, the advent of extremely efficient microprocessors has made particle filters a popular algorithm [37.70–74].

The key problem with the particle filter in the context of SLAM is that the space of maps and robot paths is huge. Suppose we have a map with 1000 features. How many particles would it take to populate that space? In fact, particle filters scale exponentially with the dimension of the underlying state space. Three or four dimensions are thus acceptable, but 100 dimensions are generally not.

The trick to make particle filters amenable to the SLAM problem goes back to [37.75, 76]. The trick was introduced into the SLAM literature in [37.77], followed by [37.78], who coined the name FastSLAM. Let us

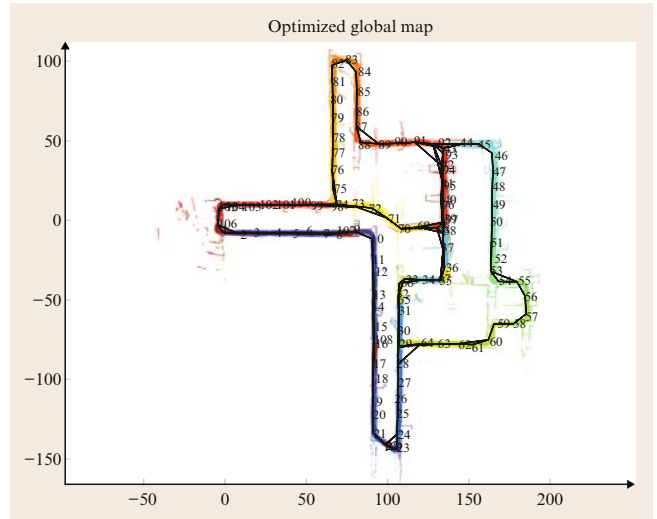


Fig. 37.7 Mine map generated by the *Atlas* SLAM algorithm by [37.26] (after Bosse et al. [37.26])

first explain the basic FastSLAM algorithm on the simplified point-landmark example, and then discuss the justification for this approach.

At any point in time, FastSLAM maintains K particles of the type:

$$X_t^{[k]}, \mu_{t,1}^{[k]}, \dots, \mu_{t,N}^{[k]}, \Sigma_{t,1}^{[k]}, \dots, \Sigma_{t,N}^{[k]}. \quad (37.16)$$

Here $[k]$ is the index of the sample. This expression states that a particle contains

- a sample path $X_t^{[k]}$
- a set of N two-dimensional Gaussians with means $\mu_{t,n}^{[k]}$ and variances $\Sigma_{t,n}^{[k]}$, one for each landmark in the environment

Here n ($1 \leq n \leq N$) is the index of the landmark. From this it follows that K particles possess K path samples. It also possesses KN Gaussians, each of which models exactly one landmark for one of the particles.

Initializing FastSLAM is simple: just set each particle's robot location to its known starting coordinates, and zero the map. The particle update then proceeds as follows.

- When an odometry reading is received, new location variables are generated stochastically, one for each of the particles. The distribution for generating those location particles is based on the motion model

$$x_t^{[k]} \sim p(x_t | x_{t-1}^{[k]}, u^t). \quad (37.17)$$

Here $x_{t-1}^{[k]}$ is the previous location, which is part of the particle. This probabilistic sampling step is easily implemented for any robot whose kinematics can be computed.

- When a measurement z_t is received, two things happen: first, FastSLAM computes for each particle the probability of the new measurement z_t . Let the index of the sensed landmark be n . Then the desired probability is defined as

$$w_t^{[k]} := \mathcal{N}(z_t | x_t^{[k]}, \mu_{t,n}^{[k]}, \Sigma_{t,n}^{[k]}). \quad (37.18)$$

The factor $w_t^{[k]}$ is called the *importance weight*, since it measures how *important* the particle is in the light of the new sensor measurement. As before, \mathcal{N} denotes the normal distribution, but this time it is calculated for a specific value, z_t . The importance weights of all particles are then normalized so that they sum to 1.

Next, FastSLAM draws with replacement from the set of existing particles a set of new particles. The probability of drawing a particle is its normalized importance weight. This step is called *resampling*. The intuition behind resampling is simple: particles for which the measurement is more plausible have a higher chance of surviving the resampling process. Finally, FastSLAM updates for the new particle set the mean $\mu_{t,n}^{[k]}$ and covariance $\Sigma_{t,n}^{[k]}$, based on the

measurement z_t . This update follows the standard EKF update rules.

This all sounds complex, but FastSLAM is quite easy to implement. Sampling from the motion model is usually straightforward, since it involves a simple kinematic calculation. Computing the importance of a measurement is, too, straightforward, especially for Gaussian measurement noise. And updating a low-dimensional particle filter is also straightforward. This makes FastSLAM one of the easiest-to-implement algorithms presently available.

FastSLAM has been shown to approximate the full SLAM posterior. The derivation of FastSLAM exploits three techniques: Rao–Blackwellization, conditional independence, and resampling. Rao–Blackwellization is the following concept. Suppose we would like to compute a probability distribution $p(a, b)$, where a and b are arbitrary random variables. The vanilla particle filter would draw particles from the joint distributions, that is, each particle would have a value for a and one for b . However, if the conditional $p(b | a)$ can be described in closed form, it is equally legitimate to just draw particles from $p(a)$, and attach to each particle a closed-form description of $p(b | a)$. This trick is known as Rao–Blackwellization, and it yields better results than sampling from the joint distribution. FastSLAM applies this technique, in that it samples from the path posterior $p(X_t^{[k]} | U_t, Z_t)$ and represents the map $p(m | X_t^{[k]}, U_t, Z_t)$ in Gaussian form.

FastSLAM also breaks down the posterior over maps (conditioned on paths) into sequences of low-dimensional Gaussians. The justification for this decomposition is subtle. It arises from a specific conditional independence assumption that is native to SLAM. Figure 37.8 illustrates the concept graphically. In SLAM, knowledge of the robot path renders all landmark estimates independent. This is easily shown for the graphical network in Fig. 37.8: we find that if we remove the path variables from Fig. 37.8 then the landmark variables are all disconnected [37.79]. Thus, in SLAM any dependence between multiple landmark estimates is mediated through the robot path. This subtle but important observation implies that, even if we used a large, monolithic Gaussian for the entire map (one per particle, of course), the off-diagonal element between different landmarks would simply remain zero. It is therefore legitimate to implement the map more efficiently, using N small Gaussians, one for each landmark. This explains the efficient map representation in FastSLAM.

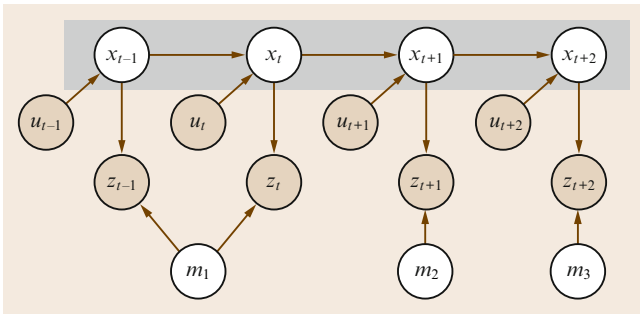


Fig. 37.8 The SLAM problem depicted as a Bayes network graph. The robot moves from location x_{t-1} to location x_{t+2} , driven by a sequence of controls. At each location x_t it observes a nearby feature in the map $m = \{m_1, m_2, m_3\}$. This graphical network illustrates that the location variables *separate* the individual features in the map from each other. If the locations are known, there remains no other path involving variables whose value is not known, between any two features in the map. This lack of a path renders the posterior of any two features in the map conditionally independent (given the locations)

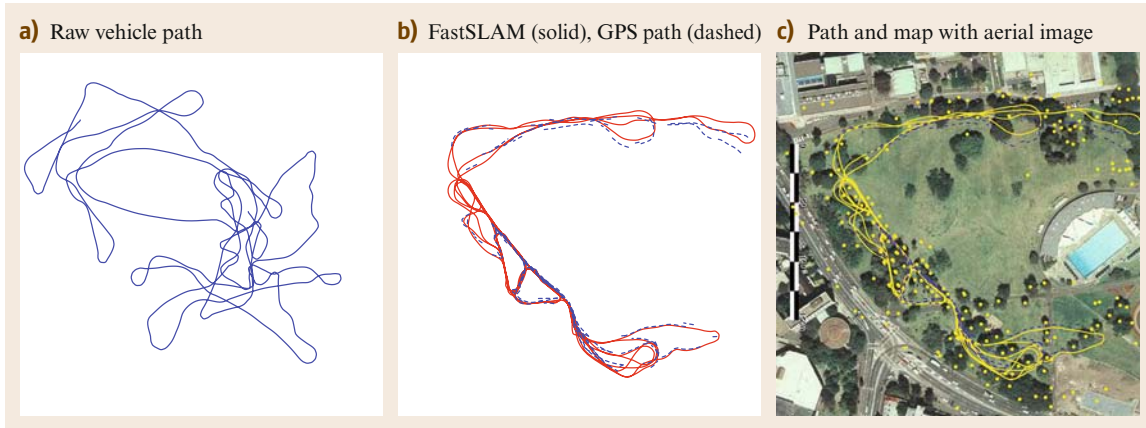


Fig. 37.9 (a) Vehicle path predicted by the odometry; (b) True path (*dashed line*) and FastSLAM 1.0 path (*solid line*); (c) Victoria Park results overlaid on aerial imagery with the **GPS** path in *blue (dashed)*, average FastSLAM 1.0 path in *yellow (solid)*, and estimated features as *yellow dots* (Data and aerial image courtesy of José Guivant and Eduardo Nebot, Australian Center for Field Robotics)

We also note that FastSLAM uses a particle filter. Derivations of the particle filter can be found in the literature referenced above. Here we note that both steps – the motion and the measurement steps – retain the property that (asymptotically) samples are drawn from the full **SLAM** posterior. This is quite easy to see for the motion update step. For the measurement step, the property is retained through resampling, which adjusts the particle population in response to the new information added by the measurement.

Figure 37.9 shows results for a point-feature problem; here the point features are the centers of tree trunks as observed by an outdoor robot. The dataset used here is known as the Victoria Park dataset [37.80]. Figure 37.9a shows that the path of the vehicle obtained by integrating the vehicle controls, without perception controls, are a poor predictor of location for this vehicle; after 30 min of driving, the estimated position of the vehicle is well over 100 m away from its **GPS** position.

The FastSLAM algorithm has a number of remarkable properties, which may not be intuitive to the untrained eye. First, it solves both the full and the on-line **SLAM** problems. Each particle has a sample of an entire path (and in fact, conditioning on the entire path is required for its mathematical derivation), but the actual update equation only uses the most recent pose. This makes FastSLAM a filter, similar to the **EKF**. Second, FastSLAM makes it easy to pursue multiple data association hypotheses. It is straightforward to make data association decisions on a per-particle basis, in-

stead of having to adopt the same hypothesis for the entire filter. While we will not give any mathematical justification, we note that the resulting FastSLAM algorithm samples the correct posteriors even for **SLAM** problems with unknown data association – something that neither of the previous two algorithms can claim. And third, FastSLAM can be implemented very efficiently: using advanced tree methods to represent the map estimates, the update can be performed in time logarithmic in the size of the map N , and linear in the number of particles M . These properties, along with the relative ease of implementation, has made FastSLAM a popular choice.

FastSLAM has been extended in great many ways. One important variant is a grid-based version of FastSLAM, in which the Gaussians are replaced by an occupancy grid map [37.81]. This variant is illustrated in Figs. 37.10 and 37.11. Figure 37.10 shows the situation just before closing a large loop. The three different particles each stand for different paths, and they also possess their own local maps. When the loop is closed, importance resampling selects those particles whose maps are most consistent with the measurement. A resulting large-scale map is shown in Fig. 37.11.

Significant extensions of the FastSLAM method can be found in [37.82, 83], whose methods *DP-SLAM* and *ancestry trees* provide efficient tree update methods for grid-based maps. The work in [37.84] provides a way to incorporate new observations into the location sampling process, based on prior work in [37.85].

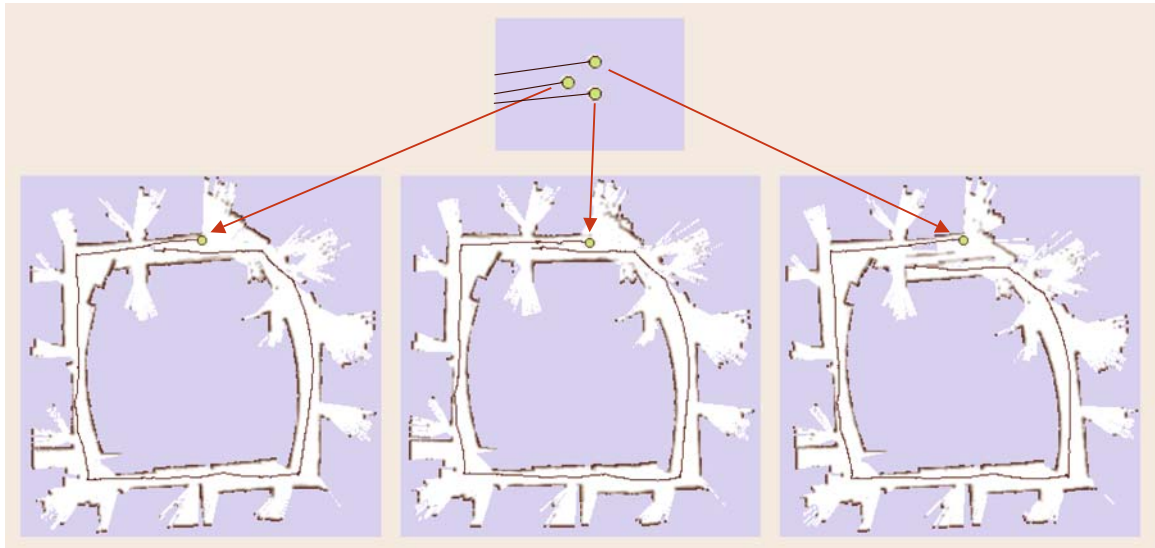


Fig. 37.10 Application of the grid-based variant of the FastSLAM algorithm. Each particle carries its own map, and the importance weights of the particles are computed based on the likelihood of the measurements given the particle's own map

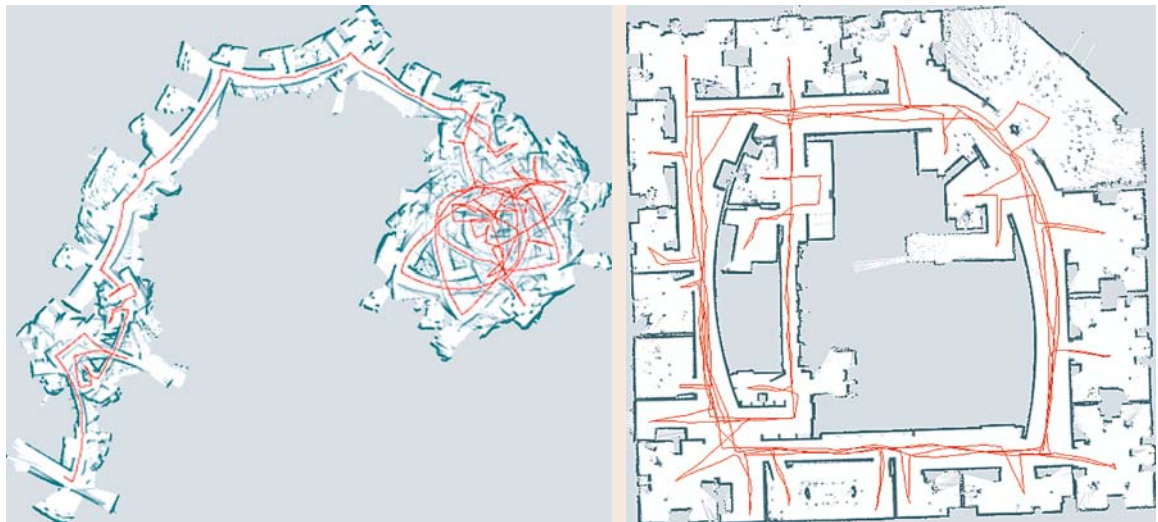


Fig. 37.11 Occupancy grid map generated from laser range data and based on pure odometry (All images courtesy of Dirk Hähnel, University of Freiburg)

37.3.4 Relation of Paradigms

The three paradigms just discussed cover the vast majority of work in the field of **SLAM**. As discussed, **EKF SLAM** comes with a computational hurdle that poses serious scaling limitations. The most promising extensions of **EKF SLAM** are based on

building local submaps; however, in many ways the resulting algorithms resemble the graph-based approach.

Graph-based methods address the full **SLAM** problem, and hence are by nature not online. They draw their intuition from the observation that **SLAM** can be modeled by a sparse graph of soft constraints,

where each constraint either corresponds to a motion or a measurement event. Due to the availability of highly efficient optimization methods for sparse non-linear optimization problems, graph-based SLAM has become the method of choice for building large-scale maps offline. Data association search is quite easily incorporated into the basic mathematical framework, and a number of search techniques exist for finding suitable correspondences. There are also extensions of the graph-based SLAM for the online SLAM problem. Those tend to remove old robot locations from the graph.

Particle filter methods sidestep some of the issues arising from the natural interfeature correlations in the map, which plagued the EKF. By sampling from robot poses, the individual landmarks in the map become

independent, and hence are decorrelated. As a result, FastSLAM can represent the posterior by a sampled robot pose, and many local, independent Gaussians for its landmarks. The particle representation of FastSLAM has a number of advantages. Computationally, FastSLAM can be used as a filter, and its update requires linear-logarithmic time where EKF needed quadratic time. Further, FastSLAM can sample over data association, which makes it a prime method for SLAM with unknown data association. On the negative side, the number of necessary particles can grow very large, especially for robots seeking to map multiple nested loops. We discussed extensions of FastSLAM that use occupancy grid maps instead of Gaussian landmarks, and showed state-of-the-art examples in large map building.

37.4 Conclusion and Future Challenges

This chapter provided a comprehensive introduction to the SLAM problem and its primary solutions. The SLAM problem was defined as the problem faced by a mobile platform roaming an unknown environment, and seeking to localize and map its environment at the same time. The chapter discussed three main paradigms in SLAM, which are based on the extended Kalman filter, graph-based sparse optimization techniques, and particle filters. It pointed out some of the advantages and disadvantages of those methods. For a more in-depth discussion, the interested reader should refer to a recent textbook covering SLAM [37.5].

Interestingly, the field of SLAM is still relatively young, and it has made enormous progress within just the past decade. In fact, nearly every method described here has been developed within the past few years. Despite all this progress, there remains a great number of open research issues that warrant future research.

In particular, SLAM techniques mostly deal with static environments, yet nearly every actual robot environment is dynamic. Early applications of SLAM methods to dynamic environments can be found in [37.86–88]. More work is needed to understand the interaction of moving and nonmoving objects in SLAM.

Most SLAM work addresses single-robot mapping, yet sometimes one is given a team of robots. Early and highly restrictive work on multirobot SLAM can be found in [37.89, 90]. More recent methods

include those in [37.91–93]. Multirobot SLAM has benefited greatly from substantial recent interest; nevertheless, the existing methods have not yet matured to a level where they can be used by nonexperts in the field.

One of the primary challenges in SLAM is to pursue significant implementations. While the theory of SLAM is now quite well developed, SLAM has not yet been used extensively in industrial or commercial applications. There exist promising prototypes, including methods for building large-scale three-dimensional (3-D) volumetric maps [37.94–97], detailed underwater reconstruction [37.31, 98, 99], and mapping of abandoned underground mines [37.57]. Nevertheless, the authors feel that more work is needed to mature the technology into industrial-strength applications.

An ultimate goal is to realize the challenge of *persistent* navigation and mapping – the capability for a robot to perform SLAM robustly for days, weeks, or months at a time with minimal human supervision, in complex, dynamic environments. Taking the limit as $t \rightarrow \infty$ poses difficult challenges to most current algorithms; in fact, most robot mapping and navigation algorithms are doomed to fail with the passage of time, as errors inevitably accrue. Research is needed to develop techniques that can recover from mistakes and enable robots to deal with changes in the environment and to recover from mistakes, enabling a long-term autonomous existence.

37.5 Suggestions for Further Reading

The following references provide an in-depth tutorial on **SLAM** and much greater depth of coverage on the details of current **SLAM** algorithms:

- H. Durrant-Whyte, T. Bailey: Simultaneous localization and mapping: Part I, *IEEE Robot. Autom. Mag.*, 99–108 (2006)
- T. Bailey, H. Durrant-Whyte: Simultaneous localization and mapping: Part II, *IEEE Robot. Autom. Mag.*, 108–117 (2006)
- S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)

References

- 37.1 K.F. Gauss: *Theoria Motus Corporum Coelestium (Theory of the Motion of the Heavenly Bodies Moving about the Sun in Conic Sections)* (Little, Brown, and Co., Republished in 1857, and by Dover in 1963, 1809)
- 37.2 G. Konecny: *Geoinformation: Remote Sensing, Photogrammetry and Geographical Information Systems* (Taylor Francis, New York 2002)
- 37.3 C. Tomasi, T. Kanade: Shape and motion from image streams under orthography: A factorization method, *Int. J. Comput. Vis.* **9**(2), 137–154 (1992)
- 37.4 S. Soatto, R. Brockett: Optimal structure from motion: Local ambiguities and global estimates, *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)* (Santa Barbara 1998) pp. 282–288
- 37.5 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT, Cambridge 2005)
- 37.6 H. Durrant-Whyte, T. Bailey: Simultaneous localization and mapping: Part I, *IEEE Robot. Autom. Mag.* (2006) pp. 99–108
- 37.7 T. Bailey, H. Durrant-Whyte: Simultaneous localization and mapping: Part II, *IEEE Robot. Autom. Mag.* (2006) pp. 108–117
- 37.8 P. Cheeseman, P. Smith: On the representation and estimation of spatial uncertainty, *Int. J. Robot. Res.* **5**, 56–68 (1986)
- 37.9 R.C. Smith, P. Cheeseman: On the representation and estimation of spatial uncertainty, *Int. J. Robot. Res.* **5**(4), 56–68 (1986)
- 37.10 R. Smith, M. Self, P. Cheeseman: Estimating uncertain spatial relationships in robotics, *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer, Berlin, Heidelberg 1990) pp. 167–193
- 37.11 P. Moutarlier, R. Chatila: An experimental system for incremental environment modeling by an autonomous mobile robot, *1st International Symposium on Experimental Robotics* (Montreal 1989)
- 37.12 P. Moutarlier, R. Chatila: Stochastic multisensory data fusion for mobile robot location and environment modeling, *5th Int. Symposium on Robotics Research* (Tokyo 1989)
- 37.13 A.M. Jazwinsky: *Stochastic Processes and Filtering Theory* (Academic, New York 1970)
- 37.14 R.E. Kalman: A new approach to linear filtering and prediction problems, *Trans. ASME J. Basic Eng.* **82**, 35–45 (1960)
- 37.15 P.S. Maybeck: The Kalman filter: An introduction to concepts. In: *Autonomous Robot Vehicles*, ed. by I.J. Cox, G.T. Wilfong (Springer, Berlin, Heidelberg 1990)
- 37.16 M. Csorba: Simultaneous Localisation and Map Building. Ph.D. Thesis (University of Oxford, Oxford 1997)
- 37.17 J. Neira, J.D. Tardós: Data association in stochastic mapping using the joint compatibility test, *IEEE Trans. Robot. Autom.* **17**(6), 890–897 (2001)
- 37.18 J. Neira, J.D. Tardós, J.A. Castellanos: Linear time vehicle relocation in SLAM, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Taiwan 2003)
- 37.19 T. Bailey: Mobile Robot Localisation and Mapping in Extensive Outdoor Environments. Ph.D. Thesis (University of Sydney, Sydney 2002)
- 37.20 G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, M. Csorba: A solution to the simultaneous localisation and map building (SLAM) problem, *IEEE Trans. Robot. Autom.* **17**(3), 229–241 (2001)
- 37.21 G. Dissanayake, S.B. Williams, H. Durrant-Whyte, T. Bailey: Map management for efficient simultaneous localization and mapping (SLAM), *Autonom. Robot.* **12**, 267–286 (2002)
- 37.22 S. Williams, G. Dissanayake, H.F. Durrant-Whyte: Constrained initialization of the simultaneous localization and mapping algorithm, *Proceedings of the Symposium on Field and Service Robotics* (Helsinki 2001)
- 37.23 J.J. Leonard, R.J. Rikoski, P.M. Newman, M. Bosse: Mapping partially observable features from multiple uncertain vantage points, *Int. J. Robot. Res.* **21**(10), 943–975 (2002)
- 37.24 A.J. Davison: Real-Time Simultaneous Localisation and Mapping with a Single Camera, *International Conference on Computer Vision* (Nice 2003) pp. 1403–1410
- 37.25 J.M.M. Montiel, J. Civera, A.J. Davison: Unified inverse depth parametrization for monocular SLAM,

- Proc. of the Robotics Science and Systems Conference (RSS06), Vol.1 (Philadelphia 2006)
- 37.26 M. Bosse, P. Newman, J. Leonard, S. Teller: Simultaneous localization and map building in large-scale cyclic environments using the atlas framework, *Int. J. Robot. Res.* **23**(12), 1113–1139 (2004)
- 37.27 J. Nieto, T. Bailey, E. Nebot: Scan-SLAM: Combining ekf-slam and scan correlation, *Proc. IEEE Int. Conf. Robotics and Automation* (Barcelona 2005)
- 37.28 J. Folkesson, H.I. Christensen: Outdoor exploration and slam using a compressed filter, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Taiwan 2003) pp. 419–427
- 37.29 J. Guivant, E. Nebot: Optimization of the simultaneous localization and map building algorithm for real time implementation, *IEEE Trans. Robot. Autom.* **17**(3), 242–257 (2001)
- 37.30 J.J. Leonard, H.J.S. Feder: A computationally efficient method for large-scale concurrent mapping and localization, *Proceedings of the Ninth International Symposium on Robotics Research*, ed. by J. Hollerbach, D. Koditschek (Salt Lake City 1999)
- 37.31 S.B. Williams, G. Dissanayake, H. Durrant-Whyte: An efficient approach to the simultaneous localisation and mapping problem, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Washington 2002) pp. 406–411
- 37.32 J.D. Tardós, J. Neira, P.M. Newman, J.J. Leonard: Robust mapping and localization in indoor environments using sonar data, *Int. J. Robot. Res.* **21**(4), 311–330 (2002)
- 37.33 S. Betgé-Brezetz, R. Chatila, M. Devy: Object-based modelling and localization in natural environments, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Osaka 1995)
- 37.34 S. Betgé-Brezetz, P. Hébert, R. Chatila, M. Devy: Uncertain map making in natural environments, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Minneapolis 1996)
- 37.35 J.E. Guivant, E.M. Nebot, J. Nieto, F. Masson: Navigation and mapping in large unstructured environments, *Int. J. Robot. Res.* **23**(4), 449–472 (2004)
- 37.36 J. Nieto, J.E. Guivant, E.M. Nebot: The hybrid metric maps (HYMMs): A novel map representation for dense SLAM, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (New Orleans 2004)
- 37.37 W. Burgard, D. Fox, H. Jans, C. Matenar, S. Thrun: Sonar-based mapping of large-scale mobile robot environments using EM, *Proceedings of the International Conference on Machine Learning (Bled 1999)*
- 37.38 H. Shatkay, L. Kaelbling: Learning topological maps with weak local odometric information, *Proceedings of IJCAI-97* (Nagoya 1997)
- 37.39 S. Thrun, D. Fox, W. Burgard: A probabilistic approach to concurrent mapping and localization for mobile robots, *Machine Learn.* **31**, 29–53 (1998), Also appeared in *Autonomous Robots* 5, 253–271 (joint issue)
- 37.40 A.P. Dempster, A.N. Laird, D.B. Rubin: Maximum likelihood from incomplete data via the EM algorithm, *J. R. Statist. Soc. Ser. B* **39**(1), 1–38 (1977)
- 37.41 H.F. Durrant-Whyte: Uncertain geometry in robotics, *IEEE Trans. Robot. Autom.* **4**(1), 23–31 (1988)
- 37.42 F. Lu, E. Milios: Globally consistent range scan alignment for environment mapping, *Autonom. Robot.* **4**, 333–349 (1997)
- 37.43 F. Dellaert: Square root SAM, *Proceedings of the Robotics Science and Systems Conference*, ed. by S. Thrun, G. Sukhatme, S. Schaal, O. Brock (Cambridge 2005)
- 37.44 T. Duckett, S. Marsland, J. Shapiro: Learning globally consistent maps by relaxation, *Proceedings of the IEEE International Conference on Robotics and Automation* (San Francisco 2000) pp. 3841–3846
- 37.45 T. Duckett, S. Marsland, J. Shapiro: Fast, on-line learning of globally consistent maps, *Auton. Robot.* **12**(3), 287–300 (2002)
- 37.46 J. Folkesson, H.I. Christensen: Graphical SLAM: A self-correcting map, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (New Orleans 2004)
- 37.47 J. Folkesson, H.I. Christensen: Robust SLAM, *Proceedings of the International Symposium on Autonomous Vehicles* (Lisboa 2004)
- 37.48 U. Frese, G. Hirzinger: Simultaneous localization and mapping – a discussion, *Proceedings of the IJCAI Workshop on Reasoning with Uncertainty in Robotics* (Seattle 2001) pp. 17–26
- 37.49 U. Frese, P. Larsson, T. Duckett: A multigrid algorithm for simultaneous localization and mapping, *IEEE Trans. Robot.* **21**(2), 196–207 (2005)
- 37.50 M. Golfarelli, D. Maio, S. Rizzi: Elastic correction of dead-reckoning errors in map building, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Victoria 1998) pp. 905–911
- 37.51 K. Konolige: Large-scale map-making, *Proceedings of the AAAI National Conference on Artificial Intelligence* (San Jose 2004) pp. 457–463
- 37.52 M. Montemerlo, S. Thrun: Large-scale robotic 3-d mapping of urban structures, *Proceedings of the International Symposium on Experimental Robotics (ISER)* (Singapore 2004)
- 37.53 Y. Liu, S. Thrun: Results for outdoor-SLAM using sparse extended information filters, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (Taiwan 2003)

- 37.54 M.A. Fischler, R.C. Bolles: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Commun. ACM* **24**, 381–395 (1981)
- 37.55 D. Hähnel, W. Burgard, B. Wegbreit, S. Thrun: Towards lazy data association in SLAM, *Proceedings of the 11th International Symposium of Robotics Research (ISRR'03)* (Sienna 2003)
- 37.56 B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, F. Savelli: Local metrical and global topological maps in the hybrid spatial semantic hierarchy, *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (New Orleans 2004)
- 37.57 S. Thrun, S. Thayer, W. Whittaker, C. Baker, W. Burgard, D. Ferguson, D. Hähnel, M. Montemerlo, A. Morris, Z. Omohundro, C. Reverte, W. Whittaker: Autonomous exploration and mapping of abandoned mines, *IEEE Robot. Autom. Mag.* **11**(4), 79–91 (2004)
- 37.58 A. Elfes: Sonar-based real-world mapping and navigation, *IEEE J. Robot. Autom.* **RA-3**(3), 249–265 (1987)
- 37.59 H.P. Moravec: Sensor fusion in certainty grids for mobile robots, *AI Mag.* **9**(2), 61–74 (1988)
- 37.60 T.M. Cover, J.A. Thomas: *Elements of Information Theory* (Wiley, New York 1991)
- 37.61 P. Newman, J.L.R. Rikoski: Towards constant-time slam on an autonomous underwater vehicle using synthetic aperture sonar, *Proceedings of the International Symposium of Robotics Research* (Sienna 2003)
- 37.62 M.A. Paskin: Thin junction tree filters for simultaneous localization and mapping, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)* (Acapulco 2003)
- 37.63 S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, A.Y. Ng: Simultaneous mapping and localization with sparse extended information filters, *Proceedings of the Fifth International Workshop on Algorithmic Foundations of Robotics*, ed. by J.-D. Boissonnat, J. Burdick, K. Goldberg, S. Hutchinson (Nice 2002)
- 37.64 E. Nettleton, S. Thrun, H. Durrant-Whyte: Decentralised slam with low-bandwidth communication for teams of airborne vehicles, *Proceedings of the International Conference on Field and Service Robotics* (Lake Yamanaka 2003)
- 37.65 P. Newman: On the Structure and Solution of the Simultaneous Localisation and Map Building Problem. Ph.D. Thesis (Australian Centre for Field Robotics, University of Sydney, Sydney 2000)
- 37.66 P.M. Newman, H.F. Durrant-Whyte: Geometric projection filter: An efficient solution to the SLAM problem, *Proc. SPIE* **4571** (2001)
- 37.67 S. Thrun, Y. Liu, D. Koller, A.Y. Ng, Z. Ghahramani, H. Durrant-Whyte: Simultaneous localization and mapping with sparse extended information filters, *Int. J. Robot. Res.* **23**(7–8), 693–716 (2004)
- 37.68 S.B. Williams: Efficient Solutions to Autonomous Mapping and Navigation Problems. Ph.D. Thesis (University of Sydney, Sydney 2001)
- 37.69 N. Metropolis, S. Ulam: The Monte Carlo method, *J. Am. Stat. Assoc.* **44**(247), 335–341 (1949)
- 37.70 D.B. Rubin: Using the SIR algorithm to simulate posterior distributions, *Bayesian Statistics 3*, ed. by M.H. Bernardo, K.M. DeGroot, D.V. Lindley, A.F.M. Smith (Oxford Univ. Press, Oxford 1988)
- 37.71 A. Doucet: *On sequential simulation-based methods for Bayesian filtering. Technical Report CUED/INFENG/TR 310* (Cambridge University, Cambridge 1998)
- 37.72 G. Kitagawa: Monte Carlo filter and smoother for non-Gaussian nonlinear state space models, *J. Comput. Graph. Statist.* **5**(1), 1–25 (1996)
- 37.73 J. Liu, R. Chen: Sequential monte carlo methods for dynamic systems, *J. Am. Stat. Assoc.* **93**, 1032–1044 (1998)
- 37.74 M. Pitt, N. Shephard: Filtering via simulation: auxiliary particle filter, *J. Am. Stat. Assoc.* **94**, 590–599 (1999)
- 37.75 D. Blackwell: Conditional expectation and unbiased sequential estimation, *Ann. Math. Statist.* **18**, 105–110 (1947)
- 37.76 C.R. Rao: Information and accuracy obtainable in estimation of statistical parameters, *Bull. Calcutta Math. Soc.* **37**, 81–91 (1945)
- 37.77 K. Murphy, S. Russell: Rao-Blackwellized particle filtering for dynamic Bayesian networks. In: *Sequential Monte Carlo Methods in Practice*, ed. by A. Doucet, N. de Freitas, N. Gordon (Springer, Berlin, Heidelberg 2001) pp. 499–516
- 37.78 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM: A factored solution to the simultaneous localization and mapping problem, *Proceedings of the AAAI National Conference on Artificial Intelligence* (Edmonton 2002)
- 37.79 J. Pearl: *Probabilistic reasoning in intelligent systems: networks of plausible inference* (Morgan Kaufmann, San Mateo 1988)
- 37.80 J. Guivant, E. Nebot, S. Baiker: Autonomous navigation and map building using laser range sensors in outdoor applications, *J. Robot. Syst.* **17**(10), 565–583 (2000)
- 37.81 D. Hähnel, D. Fox, W. Burgard, S. Thrun: A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements, *Proceedings of the Conference on Intelligent Robots and Systems (IROS)* (Las Vegas 2003)
- 37.82 A. Eliazar, R. Parr: DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)* (Acapulco 2003)

- 37.83 A. Eliazar, R. Parr: DP-SLAM 2.0, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (New Orleans 2004)
- 37.84 M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit: FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges, Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI) (Acapulco 2003)
- 37.85 R. van der Merwe, N. de Freitas, A. Doucet, E. Wan: The unscented particle filter. In: *Adv. in Neural Inform. Process. Syst.* **13** (2001)
- 37.86 D. Hähnel, D. Schulz, W. Burgard: Mobile robot mapping in populated environments, *Autonom. Robot.* **17**(7), 579–598 (2003)
- 37.87 C.-C. Wang, C. Thorpe, S. Thrun: Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (Taiwan 2003)
- 37.88 D.F. Wolf, G.S. Sukhatme: Mobile robot simultaneous localization and mapping in dynamic environments, *Autonom. Robot.* **19**(1), 53–65 (2005)
- 37.89 J.-S. Gutmann, K. Konolige: Incremental mapping of large cyclic environments, Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA) (2000)
- 37.90 E.W. Nettleton, P.W. Gibbens, H.F. Durrant-Whyte: Closed form solutions to the multiple platform simultaneous localisation and map building (slam) problem, *Sensor Fusion: Architectures, Algorithms, and Applications IV*, Vol. 4051, ed. by Bulur V. Dasarathy (Bellingham 2000) pp. 428–437
- 37.91 J. Fenwick, P. Newman, J. Leonard: Collaborative concurrent mapping and localization, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (Washington 2002)
- 37.92 I.M. Rekleitis, G. Dudek, E.E. Milios: Multi-robot collaboration for robust exploration, *Ann. Math. Artif. Intell.* **31**(1–4), 7–40 (2001)
- 37.93 S. Thrun, Y. Liu: Multi-robot SLAM with sparse extended information filters, Proceedings of the 11th International Symposium of Robotics Research (IS-RR'03) (Sienna 2003)
- 37.94 C. Frueh, A. Zakhor: Constructing 3d city models by merging ground-based and airborne views, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) (Madison 2003)
- 37.95 M. Devy, C. Parra: 3-d scene modelling and curve-based localization in natural environments, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (Leuven 1998)
- 37.96 L. Iocchi, K. Konolige, M. Bajracharya: Visually realistic mapping of a planar environment with stereo, Proceedings of the 2000 International Symposium on Experimental Robotics (Waikiki 2000)
- 37.97 S. Teller, M. Antone, Z. Bodnar, M. Bosse, S. Coorg, M. Jethwa, N. Master: Calibrated, registered images of an extended urban area, Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR) (Kauai 2001)
- 37.98 R. Eustice, H. Singh, J. Leonard, M. Walter, R. Ballard: Visually navigating the RMS Titanic with SLAM information filters, Proceedings of the Robotics Science and Systems Conference, ed. by S. Thrun, G. Sukhatme, S. Schaal, O. Brock (Cambridge 2005)
- 37.99 R. Rikoski, J. Leonard, P. Newman, H. Schmidt: Trajectory sonar perception in the ligurian sea, Proceedings of the International Symposium on Experimental Robotics (ISER) (Singapore 2004)

Behavior-Based

38. Behavior-Based Systems

Maja J. Matarić, François Michaud

Nature is filled with examples of autonomous creatures capable of dealing with the diversity, unpredictability, and rapidly changing conditions of the real world. Such creatures must make decisions and take actions based on incomplete perception, time constraints, limited knowledge about the world, cognition, reasoning and physical capabilities, in uncontrolled conditions and with very limited cues about the intent of others. Consequently, one way of evaluating intelligence is based on the creature's ability to make the most of what it has available to handle the complexities of the real world. The main objective of this chapter is to clarify *behavior-based systems* and their use in single- and multi-robot autonomous control problems and applications. The chapter is organized as follows. Section 38.1 overviews robot control, introducing behavior-based systems in relation to other established approaches to robot control. Section 38.2 follows by outlining the basic principles of behavior-based systems that make them distinct from other types of robot control architectures. The concept of basis behaviors, the means of modularizing behavior-based systems, is presented in Sect. 38.3. Section 38.4 describes how behaviors are used as building blocks for creating representations for use by behavior-based systems, enabling the robot to reason about the world and about itself in that world. Section 38.5 presents several different classes of learning methods for behavior-based systems, validated on single-robot and multi-robot sys-

38.1 Robot Control Approaches	891
38.1.1 Deliberative – Think, Then Act	892
38.1.2 Reactive – Don't Think, (Re)Act	892
38.1.3 Hybrid – Think and Act Concurrently	893
38.1.4 Behavior-Based Control – Think the Way You Act	893
38.2 Basic Principles of Behavior-Based Systems	894
38.2.1 Misconceptions	896
38.3 Basis Behaviors	897
38.4 Representation in Behavior-Based Systems	897
38.5 Learning in Behavior-Based Systems	898
38.5.1 Reinforcement Learning in Behavior-Based Systems	899
38.5.2 Learning Behavior Networks	899
38.5.3 Learning from History of Behavior Use	900
38.6 Continuing Work	902
38.6.1 Motivated Configuration of Behaviors	903
38.7 Conclusions and Further Reading	905
References	906

tems. Section 38.6 provides an overview of various robotics problems and application domains that have successfully been addressed with behavior-based control. Finally, Sect. 38.7 concludes the chapter.

38.1 Robot Control Approaches

Situated robotics deals with embodied machines in complex, challenging, often dynamically changing environments. *Situatedness* thus refers to existing in

a complex, challenging environment, and having one's behavior strongly affected by it. In contrast, robots that exist in static, unchanging environments are usu-

ally not thought to be situated. These include assembly robots operating in complex but highly structured, fixed, and strongly predictable environments, specifically engineered and controlled to enable the robot accomplish very specific tasks. The predictability and stability of the environment has a direct impact on the complexity of the robot that must operate in it; situated robots therefore present a significant challenge for the designer.

Robot control, also referred to as robot decision-making or robot computational architecture, is the process of taking information about the environment through the robot's sensors, processing it as necessary in order to make decisions about how to act, and executing actions in the environment. The complexity of the environment, i.e., the level of situatedness, has a direct impact on the complexity of control, which is, in turn, directly related to the robot's task. Control architectures are covered in Part I, Chap. 8, of the Handbook.

While there are infinitely many possible ways to program a robot, there are fundamentally four classes of robot control methods, described below.

38.1.1 Deliberative – Think, Then Act

In deliberative control, the robot uses all of the available sensory information, and all of the internally stored knowledge, to reason about what actions to take next. The control system is usually organized using a functional decomposition of the decision-making processes, consisting of a sensory processing module, a modeling module, a planning module, a value judgment module, and an execution module [38.1]. Such functional decomposition allows complex operations to be performed, but implies strong sequential interdependencies between the decision-making modules.

Reasoning in deliberative systems is typically in the form of planning, requiring a search of possible state–action sequences and their outcomes. *Planning*, a major component of artificial intelligence, is known to be a computationally complex process. The process requires the robot to perform a sequence of sense–plan–act steps (e.g., *combine the sensory data into a map of the world, then use the planner to find a path in the map, then send steps of the plan to the robot's wheels*) [38.2–4]. The robot must construct and then potentially evaluate all possible plans until it finds one that enables it to reach its goal, solve the task, or decide on a trajectory to execute. Shakey, an early mobile robot that used Strips, a general planner, is an example of such a system applied to the problem of avoiding obstacles and navigating based on vision data [38.5].

Planning requires the existence of an internal, symbolic representation of the world, which allows the robot to look ahead into the future and predict the outcomes of possible actions in various states, so as to generate plans. The internal model, thus, must be kept accurate and up to date. When there is sufficient time to generate a plan and the world model is accurate, this approach allows the robot to act strategically, selecting the best course of action for a given situation. However, being situated in a noisy, dynamic world usually makes this impossible [38.6, 7]. Today, no situated robots are purely deliberative. The advent of alternative architectures was driven by the need for faster yet appropriate action in response to the demands of complex and dynamically changing real-world environments.

38.1.2 Reactive – Don't Think, (Re)Act

Reactive control is a technique for tightly coupling sensory inputs and effector outputs, typically involving no intervening reasoning [38.8] to allow the robot to respond very quickly to changing and unstructured environments [38.9]. Reactive control is inspired by the biological notion of *stimulus–response*; it does not require the acquisition or maintenance of world models, as it does not rely on the types of complex reasoning processes utilized in deliberative control. Rather, rule-based methods involving a minimal amount of computation, and no internal representations or knowledge of the world are typically used. Reactive systems achieve rapid real-time responses by embedding the robot's controller in a collection of preprogrammed, concurrent condition–action rules with minimal internal state (e.g., *if bumped, stop; if stopped, back up*) [38.8, 10]. This makes reactive control especially well suited to dynamic and unstructured worlds where having access to a world model is not a realistic option. Furthermore, the minimal amount of computation involved means that reactive systems are able to respond in a timely manner to rapidly changing environments.

Reactive control is a powerful and effective control method that abounds in nature; insects, which vastly outnumber vertebrates, are largely reactive. However, limitations to pure reactivity include the inability to store (much if any) information or have memory or internal representations of the world [38.11], and therefore the inability to learn and improve over time. Reactive control trades off complexity of reasoning for fast reaction time. Formal analysis has shown that, for environments and tasks that can be characterized a priori, reactive controllers can be very powerful, and if properly structured,

capable of optimal performance in particular classes of problems [38.12, 13]. In other types of environments and tasks, where internal models, memory, and learning are required, reactive control is not sufficient.

38.1.3 Hybrid – Think and Act Concurrently

Hybrid control aims to combine the best aspects of reactive and deliberative control: the real-time response of reactivity and the rationality and optimality of deliberation. As a result, hybrid control systems contain two different components, the reactive/concurrent condition–action rules and the deliberative ones, which must interact in order to produce a coherent output. This is challenging because the reactive component deals with the robot’s immediate needs, such as moving while avoiding obstacles, and thus operates on a very fast time scale and uses direct external sensory data and signals. In contrast, the deliberative component uses highly abstracted, symbolic, internal representations of the world, and operates on them on a longer time scale, for example to perform global path planning or plan for high-level decision-making. As long as the outputs of the two components are not in conflict, the system requires no further coordination. However, the two parts of the system must interact if they are to benefit from each other. Consequently, the reactive system must override the deliberative one if the world presents some unexpected and immediate challenge. Analogously, the deliberative component must inform the reactive one in order to guide the robot toward more efficient and optimal trajectories and goals. The interaction of the two parts of the system requires an intermediate component, which reconciles the different representations used by the other two and any conflicts between their outputs. The construction of this intermediate component is typically the greatest challenge of hybrid system design.

Hybrid systems are referred to as using *three-layer architectures*, because of their structure, which consists of the reactive (execution) layer, intermediate (coordination) layer, and deliberative (organization/planning) layer, and which is organized according to the principle of increasing precision of control in the lower layers with decreasing intelligence [38.14]. A great deal of research has been invested into the design these components and their interactions [38.2, 15–21].

Three-layer architectures aim to harness the best of reactive control in the form of dynamic, concurrent, and time-responsive control, and the best of deliberative control, in the form of globally efficient actions over a long time scale. However, there are complex issues involved

in interfacing these fundamentally differing components and the manner in which their functionality should be partitioned is not yet well understood [38.22].

38.1.4 Behavior-Based Control – Think the Way You Act

Behavior-based control employs a set of distributed, interacting modules, called *behaviors*, that collectively achieve the desired system-level behavior. To an external observer, behaviors are patterns of the robot’s activity emerging from interactions between the robot and its environment. To a programmer, behaviors are control modules that cluster sets of constraints in order to achieve and maintain a goal [38.22, 23]. Each behavior receives inputs from sensors and/or other behaviors in the system, and provides outputs to the robot’s actuators or to other behaviors. Thus, a behavior-based controller is a structured network of interacting behaviors, with no centralized world representation or focus of control. Instead, individual behaviors and networks of behaviors maintain any state information and models.

Well-designed behavior-based systems take advantage of the dynamics of interaction among the behaviors themselves, and between the behaviors and the environment. The functionality of behavior-based systems can be said to emerge from those interactions and is thus neither a property of the robot or the environment in isolation, but rather a result of the interplay between them [38.22]. Unlike reactive control, which utilizes collections of reactive rules with little if any state and no representation, behavior-based control utilizes collections of behaviors, which have no such constraints; behaviors do have state and can be used to construct representations, thereby enabling reasoning, planning, and learning.

Each of the above approaches to robot control has its strengths and weaknesses, and all play important and successful roles in certain robot control problems and applications. Each offers interesting but different insights, and no single approach should be seen as ideal or otherwise in the absolute; rather, the choice of robot control methodology should be based on the particular task, environment, and robot.

For example, reactive control is the best choice for environments demanding immediate response, but such speed of reaction comes at the price of being myopic, not looking into the past or the future. Reactive systems are also a popular choice in highly stochastic environments, and environments that can be properly characterized so as to be encoded in a reactive input–

output mapping. Deliberative systems, on the other hand, are the only choice for domains that require a great deal of strategy and optimization, and in turn search and planning. Such domains, however, are not typical of situated robotics, but more so of scheduling, game playing, and system configuration, among others. Hybrid systems are well suited for environments and tasks where internal models and planning are needed, and the real-time demands are few, or sufficiently independent of the higher-level reasoning. Behavior-based systems, in contrast, are best suited for environments with significant dynamic changes, where fast response and adaptivity are crucial, but the ability to do some looking ahead and avoid past mistakes is required. Those capabilities are spread over the active behaviors, using active representations if necessary [38.23], as discussed later in this Chapter.

Characterizing a given robot computational architecture based on these four classes of control is often a matter of degree, as architectures attempt to combine the advantages of these paradigms, especially the responsiveness, robustness, and flexibility of the behavior-based approach with the use of abstract repre-

sentational knowledge for reasoning and planning about the world [38.22] or for managing multiple conflicting goals. For example, AuRA uses a planner to select behaviors [38.22] and 3T uses behaviors in the execution layer of a three-level hierarchical architecture [38.24]; both of these architectures dynamically reconfigure behaviors according to reasoning based on available world knowledge [38.22].

Robot control presents fundamental tradeoffs having to do with time scale of response, system organization, and modularity: thinking allows looking ahead to avoid mistakes, but only as long as sufficient, accurate, up-to-date information is available, otherwise reacting may be the best way to handle the world. As a consequence of these inherent tradeoffs, it is important to have different methodologies at our disposal rather than having to fit all controller needs into a single methodology. Selecting an appropriate control methodology and designing an architecture within it is best determined by the situation: properties of the problem, the nature of the task, the level of efficiency or optimality needed, and the capabilities of the robot, both in terms of hardware, world modeling, and computation.

38.2 Basic Principles of Behavior-Based Systems

The basic principles of behavior-based control can be summarized briefly as follows:

- Behaviors are implemented as control laws (sometimes similar to those used in control theory), either in software or hardware, as a processing element or as a procedure.
- Each behavior can take inputs from the robot's sensors (e.g., proximity sensors, range detectors, contact sensors, camera) and/or from other modules in the system, and send outputs to the robot's effectors (e.g., wheels, grippers, arm, speech) and/or to other modules.
- Many different behaviors may independently receive input from the same sensors and output action commands to the same actuators.
- Behaviors are encoded to be relatively simple, and are added to the system incrementally.
- Behaviors (or subsets thereof) are executed concurrently, not sequentially, in order to exploit parallelism and speed of computation, as well as the interaction dynamics among behaviors and between behaviors and the environment.

Behavior-based robotics was developed for situated robots, allowing them to adapt to the dynamics of real-world environments without operating upon abstract representations of reality [38.11], but also giving them more computational capability and expressivity than are available to reactive robots. Behavior-based systems maintain a tight coupling of sensing and action through behaviors, and use the behavior structure for representation and learning. Therefore, it is uncommon for a behavior to perform extensive computation or reasoning relying on a traditional world model, unless such computation can be done in a timely manner in response to dynamic and fast-changing environment and task demands.

Behaviors are designed at a variety of abstraction levels, facilitating bottom-up construction of behavior-based systems. New behaviors are introduced into the system incrementally, from the simple to the more complex, until their interaction results in the desired overall capabilities of the robot. In general, behaviors encode time-extended processes, not atomic actions that are typical of feedback control (e.g., *go-forward-by-a-small-increment* or *turn-by-a-small-angle*). As a first

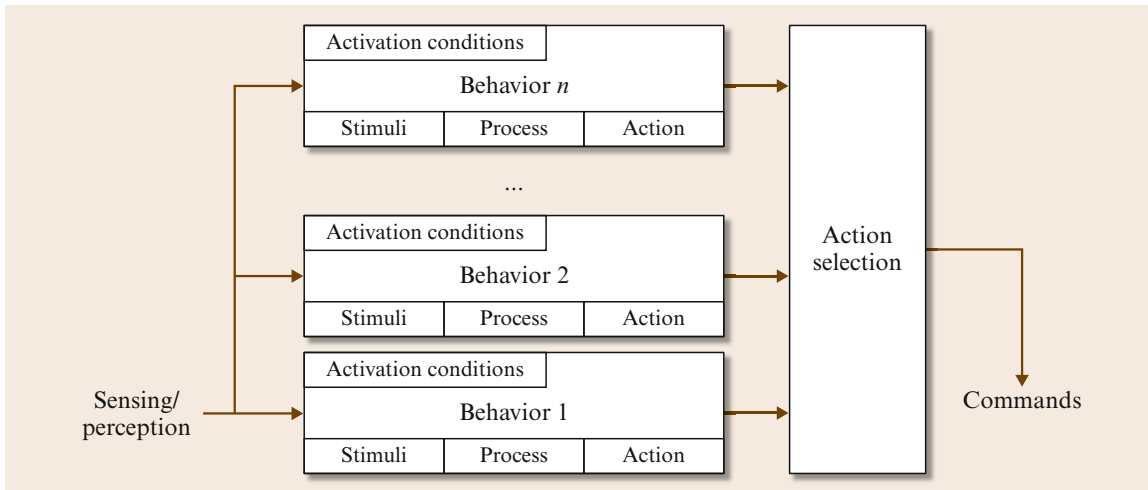


Fig. 38.1 A general schematic of one type of behavior-based systems

step, survival behaviors, such as *collision-avoidance*, are implemented. These behaviors are often reactive in nature, since reactive rules can and often do form components of simple behaviors. Figure 38.1 summarizes the general components of low-level behavior-based systems. Note that there is a distinction between *activation conditions*, which allow the behavior to generate actions, and *stimuli*, from which actions are generated.

Next, behaviors are added that provide more complex capabilities, such as *wall-following*, *target-chasing*, *homing*, *find-object*, *get-recharged*, *avoid-the-light*, *aggregate-with-group*, *pick-up-object*, *find-landmark*. Depending on the system being designed, behaviors implementing distributed representations may be added, as may be behaviors capable of learning about the world and/or the robot itself, and operating on those representations and learned information. Representation and learning are addressed in more detail in Sect. 38.4.

The interaction and integration of temporal and spatial effects is of key importance in behavior-based systems. Merely having one process controlling an actuator for predetermined intervals of time, or using as many processes as there are effectors to control them, does not suffice as the basis for behavior-based control. It is the combined effect of concurrent processes over time and driven by perception and internal states that creates the relevant behavior-based dynamics in a control system.

Behavior-based systems are typically designed so the effects of the behaviors interact largely in the en-

vironment rather than internally through the system, taking advantage of the richness of the interaction dynamics by exploiting the properties of situatedness. These dynamics are sometimes called *emergent* behaviors because they *emerge* from the interactions and are not internally specified by the robot's program. Therefore, the internal behavior structure of a behavior-based system need not necessarily mirror its externally manifested behavior. For example, a robot that flocks with other robots may not have a specific *flocking* behavior; instead, its interaction with the environment and other robots may result in flocking, although its only behaviors may be *avoid-collisions*, *stay-close-to-the-group*, and *keep-going*.

For such an approach to work, a behavior-based system must resolve the issue of choosing a particular action or behavior from multiple options, a process known as *action selection* or the *behavior coordination* problem [38.25]. This is one of the central design challenges of behavior-based systems. One approach to action selection is the use of a predefined behavior hierarchy, in which commands from the highest-ranking active behavior are sent to the actuator and all others are ignored. Numerous approaches based on other principles as well as ad hoc methods for addressing the action selection problem have been developed and demonstrated on robotic systems. These methods aim to provide increased flexibility but, in some cases, may do so at the cost of reducing the efficiency or the analyzability of the resulting control systems. Developed methods include varieties of motor schemas [38.16], command

fusion [38.26], spreading of activation through a behavior network [38.27, 28], and fuzzy logic [38.29, 30], among many others. For a survey of action selection mechanisms, see [38.31].

38.2.1 Misconceptions

Because behavior-based systems are not always simple to describe or implement, they are also often misunderstood. The most common misconception equates reactive and behavior-based systems. Historically, the advent of behavior-based systems was inspired by reactive systems, and both maintain real-time couplings between sensing and action [38.18, 32], and are structured and developed bottom-up, consisting of distributed modules. However, behavior-based systems are fundamentally *more powerful* than reactive systems, because they can store representations [38.33], while reactive systems cannot. Reactive systems are limited by their lack of internal state; they are incapable of using internal representations and learning. Behavior-based systems overcome this limitation because their underlying unit of representation, the *behavior*, can store state internally, in a distributed fashion.

The means by which state and representation are distributed in behavior-based systems is one of the sources of the flexibility of the control methodology. Representations in behavior-based systems are distributed, so as to best match and utilize the underlying behavior structure that causes the robot to act. This is how *thinking* can be organized in much the same way as *acting*. Thus if a robot needs to plan ahead, it does so in a network of communicating behaviors, rather than a single centralized planner. If a robot needs to store a large map, the map might be distributed over multiple behavior modules representing its components, such as a network of landmarks, as in [38.34], or a network of parameterized navigation behaviors, as in [38.35, 36], so that reasoning about the map/environment/task can be done in an active fashion, through using message passing within the behavior network. The planning and reasoning components of the behavior-based system use the same mechanisms as the sensing- and action-oriented behaviors, and as a result do not operate on a fundamentally different time scale and representation relative to one another. Various forms of distributed representations are used, ranging from static table structures and networks, to active procedural processes implemented within the behavior networks.

Another area of common misconception relates to the comparison between behavior-based systems and hybrid systems. Because the two use such different modularization strategies, it is often assumed that one approach (usually hybrid) has improved expressive capabilities. In fact, behavior-based and hybrid systems have the same expressive and computational capabilities: both can exploit representations and look ahead, but they do so in very different ways. This has resulted in different application domains being best suited to behavior-based versus hybrid systems. Specifically, hybrid systems dominate the domain of single-robot control, unless the task is so time-demanding that a reactive system must be used. Behavior-based systems dominate the domain of multi-robot control because the notion of collections of behaviors within the system scales well to collections of such robots, resulting in robust, adaptive group behavior [38.37, 38]. See Chap. 40 on multiple mobile robot systems for more details.

Like hybrid systems, behavior-based systems may be organized in layers, but unlike hybrid systems, the layers do not differ from each other drastically in terms of time scale and representation used. Behavior-based systems typically do not employ the hierarchical/sequential division favored by hybrid approaches. Behavior-based systems do provide both low-level control and high-level deliberation; the latter can be performed by one or more distributed representations that compute(s) over the other behaviors or modules, often directly utilizing low-level behaviors and their outputs. The resulting systems, built from the bottom up, do not divide into differently represented and independent components and consist of elements directly tied in some ways to behaviors. The power, elegance, and complexity of behavior-based systems all stem from the ways in which their constituent behaviors are designed, coordinated, and used.

To summarize briefly, behavior-based systems:

1. use behaviors as the building block of both decision-making and action execution processes;
2. use distributed parallel evaluation and concurrent control over behaviors, which take real-time inputs from sensory data and send real-time commands to effectors; and
3. have no centralized components, each module carrying out its own responsibilities. The following sections describe and illustrate in more detail how behavior-based basic principles can be used to control robots.

38.3 Basis Behaviors

The process of designing a set of behaviors for a robot is referred to as *behavior synthesis*, and is typically performed by hand, although some methods for automated synthesis behaviors have been developed and successfully demonstrated [38.39,40]. In all cases, behaviors perform a particular activity, attain a goal, or maintain some state. The notion of defining an *optimal* behavior set for a given robot or task has been considered, but it is generally accepted that such a notion is not realistic as it is dependent on too many specifics of a given system and environment that cannot currently be effectively formalized.

Matarić et al. [38.38,41] describe *basis behaviors*, also referred to as *primitive behaviors*, as a tool for structuring and thus simplifying behavior synthesis. Basis behaviors are a set of behaviors such that each is *necessary*, in the sense that each either achieves, or helps to achieve, a relevant goal that cannot be achieved without it by other members of that set. Furthermore, the basis behavior set is *sufficient* for achieving the goals mandated for the controller. The term *basis* was chosen to be indicative of the similar notion within linear algebra.

The property of necessity or parsimony is analogous to the idea of linear independence; the idea of sufficiency is similar to the linear algebraic concept of span. Basis behaviors should be simple, stable, robust, and scalable.

Another organizational principle of basis behaviors is *orthogonality*. Two behaviors are orthogonal if they do not interfere with one another, each inducing no side-effects in the other. This is often achieved by having behaviors take mutually exclusive sensory inputs. Another method is to have different behaviors control separate effectors. This form of factorization is only feasible when the robot's dynamics do not inhibit their separability. In contrast, autonomous helicopter control is an example of a highly coupled system; Saripalli et al. [38.42] demonstrated how behavior-based control can be effectively applied to robust autonomous helicopter flight.

Basis behavior design principles have been applied to single-robot as well as multi-robot behavior-based systems in a variety of applications, ranging from navigation to foraging, coordinated group movement, box pushing, and others.

38.4 Representation in Behavior-Based Systems

Embedding representation into behavior-based systems involves the challenge of conserving the basic principles of the approach at all levels of system decision-making. Combining abstract reasoning processes with behaviors must be done in a way that exploits interaction dynamics and desirable emergent system properties.

Matarić et al. [38.33,43] describe work with a robot named *Toto*, which introduced the use of distributed representation into behavior-based systems. Toto's capabilities included safe navigation, landmark detection, map learning, and path planning in the learned map representation, all within the behavior-based framework. To exploit the principles underlying behavior-based control, Toto's representation was not a centralized map. Instead, any newly discovered landmark in the environment was assigned to a new map representation behavior, which stored the landmark descriptor (type, estimated Cartesian location, and compass orientation). Whenever sensory inputs matched the landmark descriptor, the robot localized to that particular landmark and the behavior became active. The following is pseudo-code for each landmark behavior:

```

Algorithm 38.1
my-behavior-type: C
my-compass-direction: 0
my-approximate-location: (x,y)
my-approximate-length: 6.5
whenever received (input)
if input(behavior-type) = my-behavior-type
  AND
    input(compass-direction) =
      my-compass-direction
  then
    active <- true
  
```

As new landmarks are discovered, they are added to the map representation behavior network. In this way, the topology of the resulting map network is isomorphic to the topology of the network graph in the physical environment Toto has explored. The edges in the network graph are also communication links in the behavior network, allowing landmark behaviors to communicate through local message passing. Consequently, a cur-

rently active map behavior can send a message to its topological neighbor(s), thereby indicating expectation of it being the next recognized landmark and facilitating Toto's localization. Planning in the network takes place through the use of the same message-passing mechanism. The goal landmark (which could be selected by the user as part of the task, such as *go to this particular corridor* or *go to the nearest north-facing wall*), sends messages (i.e., spreads activation) from itself to its neighbors, which pass it on throughout the network. As messages are passed, the length of each landmark in the graph is accrued, thereby estimating the length of each path. The shortest path arriving at the currently active network behavior indicates the best direction to pursue toward the goal. This is equivalent to a distributed Dijkstra search. Importantly, because this search is not a static process as it would be in a centralized map representation, but an active ongoing process within a behavior map, if the robot is picked up and moved to another location, as soon as it localizes, it will continue on the optimal path to the goal; because each landmark makes a local decision as to where to go next toward the goal, no unique global path is stored in any central

location/representation. Thus, the path is constantly refreshed and updated; if any route is blocked, the link in the graph is disconnected and the shortest path is updated dynamically.

Toto exemplifies how, in behavior-based systems, representations can be stored in a distributed fashion, so as to best match the underlying behavior structure that produces the robot's external goal-driven activity. If a robot needs to make high-level decisions (such as planning ahead to a distant goal), it does so in a network of communicating behaviors, rather than a single centralized component. This results in scalable and efficient computation for the system as a whole, since the typically slower decision-making processes such as planning are distributed and modularized in a way that makes them more consistent with the time scale and representation of the rest of the system. Note the fundamental difference between this general attempt of behavior-based systems to homogenize their representation through the use of behaviors as universal modules, compared to hybrid systems, which rely on inherently different representations and time scales at different levels of the system.

38.5 Learning in Behavior-Based Systems

The ability to improve performance over time and to reason about the world, in the context of a changing and dynamic environment, is an important area of research in situated robotics. Unlike in classical machine learning, where the goal is typically to optimize performance over a long period of time, in situated learning the aim is to adapt relatively quickly, toward attaining efficiency in the light of uncertainty. Models from biology are often considered, given its properties of learning directly from environmental feedback. Variations and adaptations of machine learning, and in particular reinforcement learning, have been effectively applied to behavior-based robots, which have demonstrated learning to walk [38.44], communicate [38.45], navigate and create topological maps [38.33, 46], divide tasks [38.23, 47], behave socially [38.48], and even identify opponents and score goals in robot soccer [38.49]. Methods from artificial life, evolutionary computation/genetic algorithms, fuzzy logic, vision and learning, multi-agent systems, and many other research areas continue to be actively explored and applied to behavior-based robots as their role in animal modeling and practical applications continues to develop.

When operating in unpredictable and partially observable environments, an autonomous robot must examine the evolution of its general states, and try to capture what emerges from the interaction dynamics with its environment. Temporal integration of different types of observations is an important aspect of that capability [38.50, 51]. Work on motivational systems [38.52–56] has shown that a balance between planning and reactivity for goal management can be achieved using internal variables activated or inhibited by different factors [38.37, 57–59]. Motivations can be cyclic (e.g., circadian rhythms) or change in various temporally dependent ways [38.55]. In general, the notion of motivations is used to efficiently balance the need to adapt to the contingencies of the world and to accomplish the robot's goals.

In the following subsections, we discuss three successfully validated classes of learning approaches in behavior-based systems:

1. reinforcement learning over behaviors
2. learning behavior networks for task planning
3. learning from history of behavior use

The approaches differ in what is learned and where learning algorithms are applied, but in all cases behaviors are used as the underlying building blocks for the learning process.

38.5.1 Reinforcement Learning in Behavior-Based Systems

Behaviors are recognized as excellent substrates for speeding up reinforcement learning (RL), which is known to suffer from the curse of dimensionality. The earliest examples of RL in behavior-based systems demonstrated hexapod walking [38.44] and box-pushing [38.60]. Both decomposed the control system into a small set of behaviors, and used generalized input states, thereby effectively reducing the size of the state space. In the box-pushing example, the learning problem was broken up into modularized policies for learning separate mutually exclusive behaviors: for getting out when stuck, for finding the box when lost and not stuck, and for pushing the box when in contact with one and not stuck. The modularization into behaviors resulted in greatly accelerated as well as more robust learning.

Scaling up reinforcement learning to multi-robot behavior-based systems was explored by [38.23] and [38.61]. In multi-robot systems, the environment presents further challenges of nonstationarity and credit assignment, due to the presence of other agents and concurrent learners. The problem was studied in the context of a foraging task with four robots, each initially equipped with a small set of basis behaviors (*searching, homing, picking-up, dropping, following, avoiding*) and learning individual behavior selection policies, i.e., which behavior to execute under which conditions. Due to interference among concurrent learners, this problem could not be solved directly by standard RL. *Shaping*, a concept from psychology [38.62], was introduced; it was subsequently adopted in robot RL [38.63]. Shaping pushes the reward closer to the subgoals of the behavior, and thus encourages the learner to incrementally improve its behaviors by searching the behavior space more effectively. *Matarić* [38.61] introduced shaping through *progress estimators*, measures of progress toward the goal of a given behavior during its execution. This form of reward shaping addresses two issues associated with delayed reward: behavior termination and fortuitous reward. Behavior termination is event-driven; the duration of any given behavior is determined by the interaction dynamics with the environment, and can vary greatly. Progress estimators provide a principled

means for deciding when a behavior may be terminated even if its goal is not reached and an externally generated event has not occurred. Fortuitous reward refers to reward ascribed to a particular situation–behavior (or state–action) pair which is actually a result of previous behaviors/actions. It manifests as follows: previous behaviors lead the system near the goal, but some event induced a behavior switch, and subsequent achievement of the goal is ascribed most strongly to the final behavior, rather than the previous ones. Shaped reward in the form of progress estimators effectively eliminates this effect. Because it provides feedback during behavior execution, it rewards the previous beneficial behaviors more strongly than the final one, thus more appropriately assigning credit.

In summary, reinforcement learning has been successfully applied to behavior-based robotics, in particular at the level of behavior selection. The learning process is accelerated by the behavior structure, which provides a higher-level representation of actions and time-extended dynamics.

38.5.2 Learning Behavior Networks

The modularization of behavior-based systems as networks of behaviors allows for learning to be applied at the network level as well. *Nicolescu et al.* [38.35,36] developed the notion of *abstract behaviors*, which separate the activation conditions of a behavior from its output actions (so-called *primitive behaviors*, which share the same principles as basis behaviors described Sect. 38.3); this allows for a more general set of activation conditions to be associated with the primitive behaviors. While this is not necessary for any single task, it is what provides generality to the representation. An abstract behavior is a pairing of a given behavior's activation conditions (i.e., preconditions) and its effects (i.e., postconditions); the result is an abstract and general operator much like those used in classical deliberative systems (Fig. 38.2). Primitive behaviors, which typically consist of a small basis set, may involve one or an entire collection of sequential or concurrently executing behaviors.

Networks of such behaviors are then used to specify strategies or general *plans* in a way that merges the advantages of both abstract representations and behavior-based systems. The nodes in the networks are abstract behaviors, and the links between them represent precondition and postcondition dependencies. The task plan or strategy is represented as a network of such behaviors. As in any behavior-based system, when the conditions of a behavior are met, the behavior is acti-

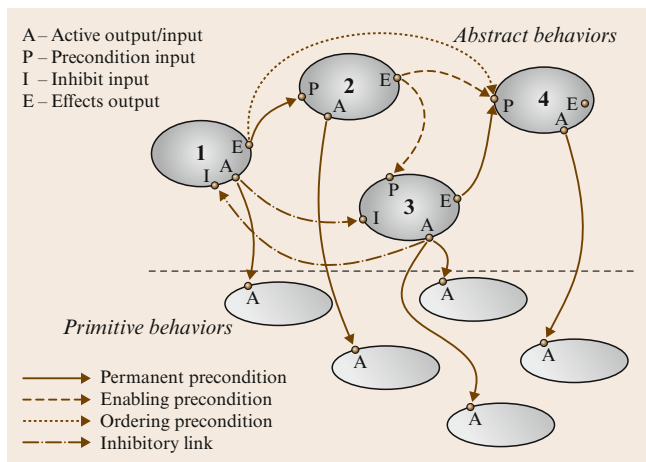


Fig. 38.2 Behavior network

vated. Analogously, when the conditions of an abstract behavior are met, the behavior activates one or more primitive behaviors which achieve the effects specified in its postconditions. The network topology at the abstract behavior level encodes any task-specific behavior sequences, freeing up the primitive behaviors to be reused for a variety of tasks. Thus, since abstract behavior networks are computationally lightweight, solutions for multiple tasks can be encoded within a single system and dynamically switched.

Nicolescu et al. [38.35, 36] introduced a means for automatically generating such networks offline as well as at runtime. The latter enables a learning robot to acquire task descriptions dynamically, while observing its environment, which can include other robots and/or a teacher. The methodology was validated on a mobile robot following a human and acquiring a representation of the human-demonstrated task by observing the activation of its own abstract behavior pre- and postconditions, thus resulting in a new abstract behavior network representing the demonstrated task [38.64]. The robot was able to acquire novel behavior sequences and combinations (i. e., concurrently executing behaviors), resulting in successful learning of tasks involving visiting various targets in particular order, picking up, transporting, and delivering objects, dealing with barriers, and maneuvering obstacle courses in specific ways.

38.5.3 Learning from History of Behavior Use

Most deliberative approaches derive knowledge for reasoning about the world from sensor inputs and actions

taken by the robot. This results in complex state-space representations of the world and does not take into consideration the context in which these sensations/actions are taken. As already discussed, behaviors, which are readily used as low-level control blocks driven by what is experienced from the interactions with the environment, can also serve as an abstract representation to model those interactions. One approach is to use history information [38.65], i. e., to explicitly take into consideration the time sequence of observations in order to make a decision. Applying this idea to behavior-based systems, by knowing the purpose of each of the behaviors and by observing their history of use, the robot can reason and ground its intentions in what it is experiencing in its operating environment. The concept of abstract behavior is exploited here as well, to activate behaviors and as a representation on which to learn.

Learning from history has been validated in behavior-based systems capable of making the robot change its behavior selection strategy for foraging colored objects (blocks) to a homing region in nonstationary, dynamically changing environments involving multiple concurrently learning robots [38.66, 67]. In that foraging task, the robot is given two subtasks: search for a block (searching task), and bring it to the home region (homing task). The robot is given behaviors to accomplish these tasks: one behavior for the searching task, called *searching-block*, and two for the homing task, *homing* and *drop-block*. A *velocity control* behavior is also used in both of these tasks to make the robot move. All these behaviors are referred to as *task behaviors*. Conditions for activating task behaviors are preprogrammed based on the presence or absence of a block in front of the robot and the proximity of the home region.

The robot also needs to navigate safely in the environment. In this approach, an *avoidance* behavior is activated unless the robot is near home and carrying a block; otherwise it is disabled, to allow the robot to approach the home region. This type of behavior, used for handling harmful situations and interference while accomplishing a task, is referred to as a *maintenance behavior*. The designer determines the conditions in which maintenance behaviors should be used, but cannot indicate when they will occur during the task, as that is tied to the dynamics of the interaction between the robot and its environment.

The robot learns to use *alternative behaviors* (*follow-side*, *rest*, and *turn-randomly*), which introduce variety into its action repertoire that changes the way it accomplishes its tasks. In contrast to other types of behaviors,

these have no a priori activation conditions; the objective is to allow the robot to learn when to activate these behaviors according to past experiences, for some pre-set periods of time, when it is experiencing interference in accomplishing its task. Figure 38.3 illustrates how the behaviors are prioritized using a fixed suppression mechanism, similar to the subsumption architecture [38.9] but with the difference that the activated behaviors, i. e., those allowed to issue outputs, change dynamically. Following behavior-based principles, an activated behavior may or may not be used to control the robot, depending on the sensory conditions it monitors and the arbitration mechanism. An activated behavior is used only when it provides commands that actually control the robot. Whenever a behavior is used, its corresponding symbol is sent to the interaction model, generating the sequence of behaviors used over time. Separate learning trees are used for each task; determining which one to use is done based on the activated task behavior.

The algorithm uses a tree structure to store the history of behavior use. The upper part of Fig. 38.3 shows a typical tree with nodes storing the behavior (H for use by *homing* and *drop-block*) used for controlling the robot while accomplishing its task, and n , the number of times a transition between the node itself and its successor has been made (as observed from behavior use). Initially, the tree for a particular task is empty and is incrementally constructed as the robot goes about its task. Leaf nodes are labeled with E (for end-node) and store the total *performance* of the particular tree path. Whenever a path is completely reused, when the same sequence of behaviors is used, the E node is updated with the average of the stored and current performances for recent trials.

Learning is done through reinforcement. Depending on the domains and the tasks, a variety of factors can be used to evaluate performance, and different metrics can be used with this learning algorithm. To see how far the idea of self-evaluation and learning by observing behavior use can be taken, the evaluation function used here is not based on characteristics about the environment or the task. Instead, it is based on the amount of time behaviors are used. Comparison between the time spent using behaviors associated with the tasks and the time spent exploiting maintenance behaviors is used to derive the evaluation criterion. Consequently, behavioral selection strategy is derived from what can be learned from the experiences of the robot in its environment, without having to characterize a priori what can be considered to be optimal operating conditions in the environment.

Using the tree and the evaluation function, the algorithm has two options for using a maintenance behavior:

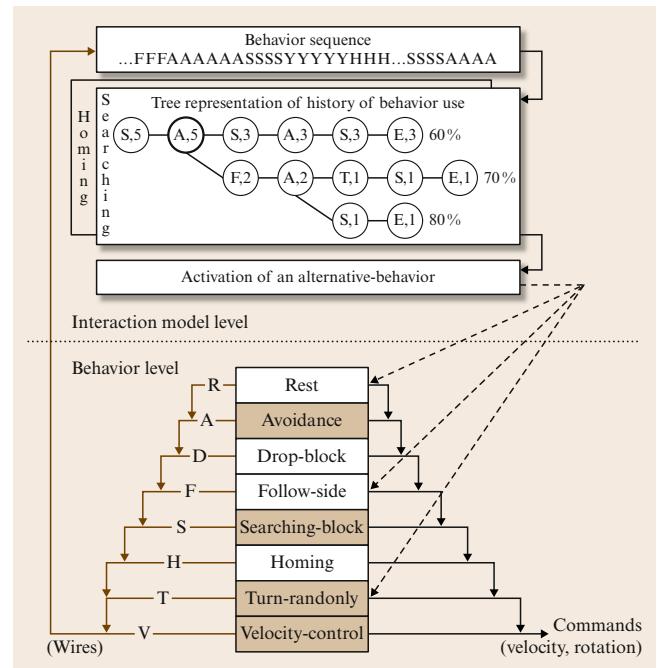


Fig. 38.3 Organization of the behavior level and the interaction model level. Behavior on colored background represent an example of activated behaviors for the searching task, with *Turn-randomly* as a chosen alternative behavior. For clarity, sensory inputs to behaviors are not shown

1. not to make any changes in its active behavior set (the *observe* option)
2. to activate an alternative behavior

The sequence of nodes in a tree path characterizes the interactions experienced by the robot in the world. Different selection criteria can be used by comparing the performance at the current position in the tree with the expected performance, following an exploration (to learn the effects of alternative behaviors) then exploitation (to exploit what was learned in previous trials) strategy. The expected performance of a given node is the sum of the stored end-node performances in its subpaths, multiplied by the frequency of use of the subpaths relative to the current position in the tree. Finally, since this algorithm is used in noisy and nonstationary conditions, deleting paths is necessary to keep the interaction model up to date. This is done by keeping only a fixed number of the most recent paths in the tree.

Results obtained with this approach show that the robot is able to learn unanticipated strategies (like resting in front of a static obstacle to increase the turning angle

and locate a target) and original ones (like yielding when close to other robots or following walls when the center of the pen is crowded). Developing such capabilities

is important in general, because it makes it possible for robots to learn in nonstationary environments, which are common in the world applications.

38.6 Continuing Work

Behavior-based robots have demonstrated various standard capabilities, including obstacle avoidance, navigation, terrain mapping, following, chasing/pursuit, object manipulation, task division and cooperation, and learning maps, navigation and walking. They have also demonstrated some novel applications like large-scale group behaviors including flocking, foraging, and soccer playing, human–robot interaction, and modeling insect and even human behavior [38.68–70]. Behavior-based methods span mobile robots, underwater vehicles, planetary rovers for space exploration, interactive and social robots, as well as robots capable of grasping, manipulation, walking, running, and many others. Broadly adopted consumer market products, such as the iRobot Roomba Robotic Floorvac [38.71], also use behavior-based control, demonstrating its broad applicability.

The use of behavior-based architectures for robot control has evolved from single-system implementations to approaches that combine forms of learning, state estimation, and distributed computation. Behaviors have been combined with a fuzzy inference system for indoor navigation using mobile robots [38.29, 30, 72, 73], where a command fusion module acts as an arbiter that combines multiple fuzzy behavior outputs into a single control signal. This strategy ensures the robot is capable of making inferences in the face of uncertainty.

Behavior-based methods have been employed in multi-robot systems from the outset [38.38]. More recently, multi-robot researchers have begun to consider tasks requiring tightly coupled cooperation; see [38.74] for an overview and discussion. Such tasks typically require low-level sensor-sharing and/or for higher-level explicit coordination. Behavior-based controllers have been developed and extended in order to address these challenges. For example, Parker et al. [38.75] considered reusable behavior units that can be automatically redistributed for low-level information processing. Wergler et al. [38.76] described broadcast of local eligibility (BLE) to enable higher-level group behaviors by allowing communications to influence each robot's local action selection mechanism. Gerkey et al. [38.77, 78] demonstrated scalable and efficient market-based co-

ordination algorithms for multi-robot coordination in a variety of tasks, including tightly coupled ones (e.g., box-pushing [38.79]).

Several researchers have shown that behavior-based controllers allow for sophisticated coordination through a utility-centered model of the collective task. Behaviors use this representation to produce actions that consider each robot's impact on the performance of the group as a whole. Iocchi et al. [38.80] have shown this in heterogeneous multi-robot system, while Batalin et al. [38.81] demonstrated that complex, interrelated and dynamic tasks can be performed in coordinated ways in robots with behavior-based controllers interacting with a sensor network. Stroupe et al. [38.82] considered a mapping task and developed move value estimation for robot teams (MVERT), essentially a behavior-based method for maximizing group knowledge.

Because behavior-based methods lend itself naturally to multi-robot control problems, they have had a significant influence on that field of research. Simmons et al. [38.83] described a hybrid architecture designed for group-level coordination which employs behaviors as a method for organizing low-level safety-critical controller code. Behaviors have also been used to structure controllers and networked communication in minimalist systems [38.84]. Some multi-robot research with a control-theoretic flavor has used separate executable processes that can be switched on and off dynamically depending on task constraints, in a manner very similar to behavior-based control [38.85].

Learning by demonstration has also been combined with behavior-based systems to produce one-shot learning and teaching mechanisms for robot control [38.35]. Furthermore, this type of behavior-based architecture has been used to learn ship navigation strategies [38.86]; during a learning phase, an instructor selects behaviors for a ship to execute in order to reach a specific goal, and a subsequent offline stage then generates dependency links between the behaviors that it witnessed during the learning phase.

Another form of learning by demonstration has used probabilistic methods to select and combine multiple behaviors [38.87]. There, the problem of learning what

behaviors to execute during autonomous navigation is treated as a state estimation problem. During a learning phase, the robot observes commands used by a teacher. A particle filter then fuses the control commands that were demonstrated by the teacher to estimate behavior activation. The method produces a robust controller that is well suited for dynamic environments.

Behavior-based architectures have also been used in complex vision systems, for identification rather than control. In those contexts, each behavior represents a small unit of visual computation, such as frame differencing, motion detection, and edge detection, resulting in biologically inspired vision and attention behavior [38.88, 89].

Behavior-based architectures have also been developed for the control and learning in humanoid robots. Figure 38.4 illustrates two robots used in ongoing projects. In the past, the Cog project demonstrated behavior-based control for articulated manual and eye–hand coordination [38.90, 91]. Edsinger [38.92] developed a lightweight behavior architecture for organizing perception and control of Domo. The architecture allows for specification of time-contingent behaviors and distributed computation, resulting in a real-time controller that allows Domo to operate in an environment with humans. Kismet [38.93] involved several behavior-based systems that controlled perception, motivation, attention, behavior, and movement in a human–robot interaction (HRI) context. Each behavior represented Kismet’s individual drives and motivations. Situated modules and episode rules were introduced as part of an HRI architecture by Ishiguro et al. [38.94] and Kanda et al. [38.95]; they employed an ordered set of general-use situated modules and behaviors that are contingent

on a condition, used in sequences defined by a set of task-specific episode rules.

38.6.1 Motivated Configuration of Behaviors

Designing robots capable of safely and effectively interacting with humans in real-world settings is one of the ultimate challenges in robotics. Representation and abstract reasoning may become necessary to keep up with and adapt to the inherently complex properties of such environments and tasks.

One possible solution currently being developed and validated is to add to a behavior-based architecture the idea of intentionally activating, monitoring, and configuring behaviors. Behaviors are considered basic control components that are selected and modified according to the intentions of the robot [38.57, 96, 97]. Such intentions can also comply with the distributed philosophy of behavior-based system by having multiple sources of influences. The motivated behavioral architecture (MBA) uses motivational modules to derive the robot’s intentions.

Motivational modules recommend the use or the inhibition of tasks to be accomplished by the robot. Tasks are data structures associated with particular configuration and activation of one or more behaviors. The processing of the robot’s intentions is done through the *dynamic task workspace* (DTW), which organizes tasks in a tree-like structure according to their interdependencies, from high-level abstract tasks (e.g., *deliver message*) to primitive behavior-related tasks (e.g., *avoid obstacles*). Through the DTW and task representation, motivational modules exchange information asynchronously on how to activate, configure, and monitor behaviors, by submitting modification requests and queries or subscribe to events regarding the task’s status. With multiple tasks being issued by the motivational modules, the *behavior activation and configuration* module determines which behaviors are to be activated according to recommendations made by motivational modules, with or without a particular configuration (e.g., a destination to go to). A recommendation can either be negative, neutral, or positive, or take on real values within this range to reflect the desirability of the robot to accomplish specific tasks. The decisional process implemented in the *behavior activation and configuration* module (which can be based on different methods of action selection, as set by the designer) takes these information to activate behaviors. Activation values reflect the resulting robot’s intentions

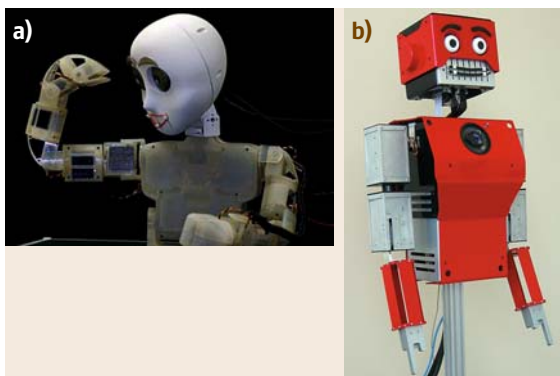


Fig. 38.4a,b Examples of behavior-based control systems used in human–robot interaction: USC Bandit (a) and University of Sherbrooke Melvin (b)

derived from interactions between the motivational modules. Behavior use and other information (useful for task representation and for monitoring how the behaviors are used by the robot to interact with the world) are also communicated through the *behavior activation and configuration* module.

Motivational modules are categorized as instinctual, rational, or emotional. Instinctual motivations provide basic operation of the robot through the use of simple rules. Rational motivations are related to cognitive processes, such as navigation and planning. Emotional motivations monitor conflicting or transitional situations between tasks, such as changes in commitments the robot establishes with other agents (humans or robot) in its environment. The manifested robot behavior can thus be appropriately influenced by direct perception, by reasoning, or by managing commitments and choices. By distributing motivational modules and distinguishing their roles, it is possible to more efficiently exploit and combine the various influences on the tasks the robot must accomplish. It is also by exchanging information through the DTW that motivational modules are kept generic and independent from each other, allowing for behavior configurations to arise in a distributed fashion, based on the capabilities available to the robot. For instance, one instinctual motivation may monitor the robot's energy level to issue a recharging task in the DTW, which activates a *recharge* behavior that would make the robot detect and dock to a charging station. Meanwhile, if the robot knows where it is and can determine a path to a nearby charging station,

a path-planning rational motivation can add a subtask of navigating to this position, using a *goto* behavior. Otherwise, the *recharge* behavior will at least allow the robot to recharge opportunistically, when it perceives a charging station.

The described architecture was used to integrate a number of intelligent decision-making capabilities on a robot named Spartacus, shown in Fig. 38.5. In the American Association for Artificial Intelligence (AAAI) mobile robot competition, Spartacus was used to demonstrate how a behavior-based system could integrate planning and sequencing tasks under temporal constraints and spatial localization capabilities, using a previously generated metric map. The system also used behavioral message reading [38.98], sound processing capabilities with an eight-microphone system for source localization, tracking, real-time sound separation [38.99, 100], and a touch screen interface to allow the robot to acquire information about where it is in the world, what it should do, and how it should do it [38.101, 102].

Figure 38.6 illustrates the navigation portion of the architecture implemented on Spartacus. Behavior action selection scheme used is priority based, and behavior recommendation and activation are binary. The behaviors used are: *stop/rest*, stopping the robot when the emergency stop or interacting with people using the graphical interface is required; *avoid*, making the robot move safely in the environment; *obey*, executing vocal navigation requests; *recharge*, stopping the robot while waiting to be connected to a charging station; *goto*, directing the robot to a specific location; *follow-sound*, making the robot follow an audio source; *follow-wall*, making the robot follow a wall (or corridor) when detected, otherwise generating a constant forward velocity. Only instinctual and rational motivations are implemented in this version, with rational motivations having greater priority over instinctual ones in case of conflicts. For instinctual motivations, the *task selector* selects one high-level task when none has yet been prioritized. For instance, between tasks that require the robot to go to a specific location, this motivation selects the task where the location is physically closest to the robot. *Safe navigation* urges the robot to maintain its physical integrity by recommending obstacle avoidance. For the rational motivations, *planner* determines which primitive tasks and sequences thereof are necessary to accomplish high-level tasks under temporal constraints and limited capabilities. The first implementation was a simple reactive planning module that interleaves planning and execution [38.103], as in [38.104] and [38.105]. *Naviga-*

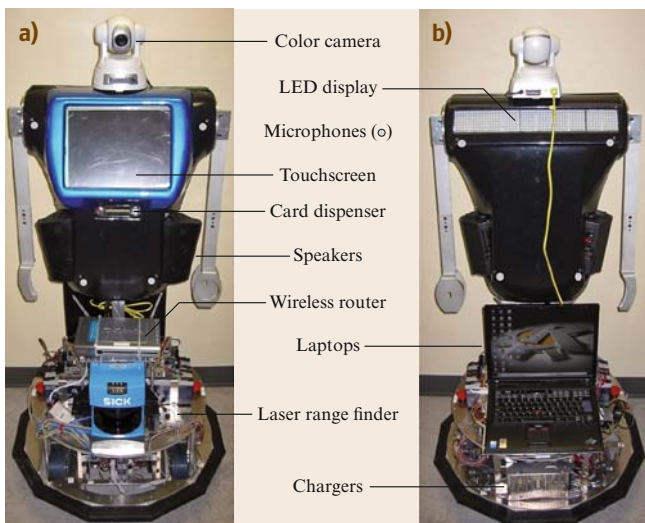


Fig. 38.5a,b Spartacus (front view (a), back view (b))

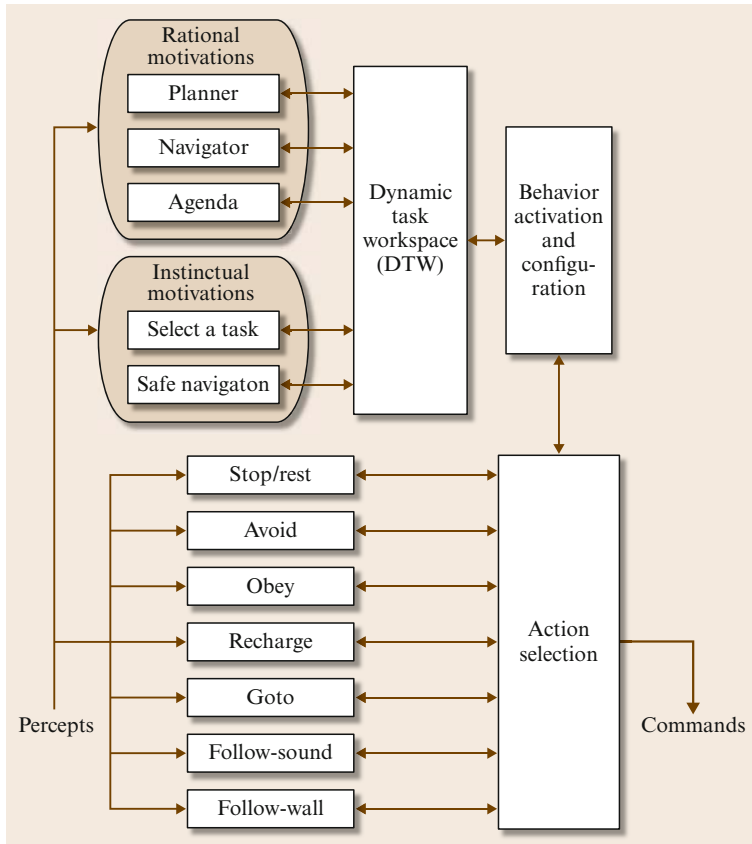


Fig. 38.6 Behavior-based architecture with distributed motivational modules

tor determined the path to a specific location, as required for tasks in the DTW. *Agenda* generated predetermined sequences of tasks to accomplish.

The underlying principles of the described architecture have also been applied to robots with different capabilities, such as a robot that uses activation variables, topological localization and mapping, and fuzzy behaviors to explore and characterize an environ-

ment [38.57, 97], and on an autonomous rolling robot that only uses simple sensors and a microcontroller to generate purposeful movements used in a study regarding interaction with toddlers [38.106, 107]. The MBA architecture is now being used on robots with increasing perceptual and action capabilities, in an attempt to provide robots with the necessary skills to be useful and efficient in daily life.

38.7 Conclusions and Further Reading

This chapter has described behavior-based control, a methodology for single- and multi-robot control aimed at situated robots operating in unconstrained, challenging, and dynamic conditions in the real world. While inspired by the philosophy of reactive control, behavior-based systems are fundamentally more expressive and powerful, enabling representation, planning, and learning capabilities. Distributed

behaviors are used as the underlying building blocks for these capabilities, allowing behavior-based systems to take advantage of dynamic interactions with the environment rather than rely solely on explicit reasoning and planning. As the complexity of robots continues to increase, behavior-based principles and their applications in robot architectures and deployed systems will evolve as well, demonstrating

increasingly higher levels of situated intelligence and autonomy.

Interested readers can find more information regarding behavior-based systems in other chapters of this

Handbook, as well as in Brooks [38.108], Arkin [38.22], in artificial intelligence and robotics textbooks [38.109, 110], and in introductory textbooks on mobile robot control [38.111–113].

References

- 38.1 J.S. Albus: Outline for a theory of intelligence, IEEE Trans. Syst. Man Cybernet. **21**(3), 473–509 (1991)
- 38.2 G. Girald, R. Chatila, M. Vaisset: *An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots*, *Proceedings First International Symposium on Robotics Research* (MIT Press, Cambridge 1983)
- 38.3 H. Moravec, A. Elfes: High resolution maps from wide angle sonar, *Proceedings IEEE International Conference on Robotics and Automation* (1995)
- 38.4 J. Laird, P. Rosenbloom: *An Investigation into Reactive Planning in Complex Domains*, *Proceedings Ninth National Conference of the American Association for Artificial Intelligence* (MIT Press, Cambridge 1990) pp. 1022–1029
- 38.5 N. J. Nilsson: Shakey the Robot, Technical Report (325) (SRI International, 1984)
- 38.6 S.J. Rosenschein, L.P. Kaelbling: A situated view of representation and control, *Artif. Intell.* **73**, 149–173 (1995)
- 38.7 R.A. Brooks: Elephants don't play chess. In: *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back* (The MIT Press, Bradford Book 1990) pp. 3–15
- 38.8 R. Brooks, J. Connell: Asynchronous distributed control system for a mobile robot, *Proceedings SPIE Intelligent Control and Adaptive Systems* (1986) pp. 77–84
- 38.9 R.A. Brooks: A robust layered control system for a mobile robot, *IEEE J. Robot. Autom.* **RA-2**(1), 14–23 (1986)
- 38.10 P.E. Agre, D. Chapman: Pengi: An implementation of a theory of activity, *Proceedings Sixth National Conference of the American Association for Artificial Intelligence* (1987) pp. 268–272
- 38.11 R.A. Brooks: Intelligence without representation, *Artif. Intell.* **47**, 139–159 (1991)
- 38.12 M. Schoppers: Universal plans for reactive robots in unpredictable domains, *Proceedings International Joint Conference on Artificial Intelligence* (1987) pp. 1039–1046
- 38.13 P.E. Agre, D. Chapman: What are plans for?. In: *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Bradford Book 1990) pp. 17–34
- 38.14 G.N. Saridis: Intelligent robotic control, *IEEE Trans. Autom. Contr.* **AC-28**(5), 547–557 (1983)
- 38.15 R.J. Firby: An investigation into reactive planning in complex domains, *Proceedings AAAI Conference* (1987) pp. 202–206
- 38.16 R. Arkin: Towards the unification of navigational planning and reactive control, *Proceedings American Association for Artificial Intelligence, Spring Symposium on Robot Navigation* (1989) pp. 1–5
- 38.17 C. Malcolm, T. Smithers: Symbol grounding via a hybrid architecture in an autonomous assembly system. In: *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Bradford Book 1990) pp. 123–144
- 38.18 J.H. Connell: SSS: A hybrid architecture applied to robot navigation, *Proceedings IEEE International Conference on Robotics and Automation* (1992) pp. 2719–2724
- 38.19 E. Gat: Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots, *Proceedings National Conference on Artificial Intelligence* (1992) pp. 809–815
- 38.20 M. Georgeoff, A. Lansky: Reactive reasoning and planning, *Proceedings Sixth National Conference of the American Association for Artificial Intelligence* (1987) pp. 677–682
- 38.21 B. Pell, D. Bernard, S. Chien, E. Gat, N. Muscettola, P. Nayak, M. Wagner, B. Williams: An autonomous spacecraft agent prototype, *Autonom. Robot.* **1-2**(5), 1–27 (1998)
- 38.22 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Bradford Book 1998)
- 38.23 M.J. Mataric: Reinforcement learning in the multi-robot domain, *Autonom. Robot.* **4**(1), 73–83 (1997)
- 38.24 P. Bonasso, R.J. Firby, E. Gat, D. Kortenkamp, D.P. Miller, M.G. Slack: Experiences with an architecture for intelligent reactive agents, *Proceedings International Joint Conference on Artificial Intelligence* (1995)
- 38.25 P. Pirjanian: Multiple objective behavior-based control, *Robot. Autonom. Syst.* **31**(1–2), 53–60 (2000)
- 38.26 D. Payton, D. Keirsey, D. Kimble, J. Krozel, J. Rosenblatt: Do whatever works: A robust approach to fault-tolerant autonomous control, *Appl. Intell.* **2**(3), 225–250 (1992)
- 38.27 P. Maes: Situated agents can have goals. In: *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, ed. by P. Maes (MIT Press, Bradford Book 1990) pp. 49–70
- 38.28 P. Maes: The dynamics of action selection, *Proceedings International Joint Conference on Artificial Intelligence* (1989) pp. 991–997

- 38.29 A. Saffiotti: The uses of fuzzy logic in autonomous robot navigation, *Soft Comput.* **1**, 180–197 (1997)
- 38.30 F. Michaud: Selecting behaviors using fuzzy logic, *Proceedings IEEE International Conference on Fuzzy Systems* (1997) pp.
- 38.31 P. Pirjanian: Behavior coordination mechanisms—State-of-the-art, (Technical Report IRIS-99-375, University of Southern California, Institute of Robotics and Intelligent Systems) (1999)
- 38.32 E. Gat: On three-layer architectures. In: *Artificial Intelligence and Mobile Robotics*, ed. by D. Kortenkamp, R. Bonasso, R. Murphy (MIT/AAAI Press, Cambridge 1998)
- 38.33 M.J. Mataric: Integration of representation into goal-driven behavior-based robots, *IEEE Trans. Robot. Autom.* **8**(3), 304–312 (1992)
- 38.34 M.J. Mataric: *Navigating with a Rat Brain: A Neurobiologically-Inspired Model for Robot Spatial Representation, From Animals to Animats. Proceedings First International Conference on Simulation of Adaptive Behaviors* (MIT Press, Bradford Book 1990) pp. 169–175
- 38.35 M. Niclescu, M.J. Mataric: Experience-based representation construction: Learning from human and robot teachers, *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems* (2001) pp. 740–745
- 38.36 M. Niclescu, M.J. Mataric: A hierarchical architecture for behavior-based robots, *Proceedings International Joint Conference on Autonomous Agents and Multiagent Systems* (2002)
- 38.37 L.E. Parker: ALLIANCE: An architecture for fault tolerant multirobot cooperation, *IEEE Trans. Robot. Autom.* **14**(2), 220–240 (1998)
- 38.38 M.J. Mataric: Designing and understanding adaptive group behavior, *Adapt. Behav.* **4**(1), 50–81 (1995)
- 38.39 O.C. Jenkins, M.J. Mataric: Deriving action and behavior primitives from human motion data, *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems* (2002) pp. 2551–2556
- 38.40 O.C. Jenkins, M.J. Mataric: Automated derivation of behavior vocabularies for autonomous humanoid motion, *Proceedings Second International Joint Conference on Autonomous Agents and Multiagent Systems* (2003)
- 38.41 M.J. Mataric: *Designing Emergent Behaviors: From Local Interactions to Collective Intelligence, From Animals to Animats 2. Proceedings Second International Conference on Simulation of Adaptive Behaviors*, ed. by J.-A. Meyer, H. Roitblat, S. Wilson (MIT Press, Cambridge 1992) pp. 432–441
- 38.42 S. Saripalli, D.J. Naffin, G.S. Sukhatme: *Autonomous Flying Vehicle Research at the University of Southern California, Multi-Robot Systems: From Swarms to Intelligent Automata, Proceedings of the First International Workshop on Multi-Robot Systems*, ed. by A. Schultz, L.E. Parker (Kluwer, Dordrecht 2002) pp. 73–82
- 38.43 M.J. Mataric: Behavior-based control: Examples from navigation, learning, and group behavior, *J. Exp. Theor. Artif. Intell.* **9**(2–3), 323–336 (1997)
- 38.44 P. Maes, R.A. Brooks: Learning to coordinate behaviors, *Proceedings Eighth National Conference on Artificial Intelligence AAAI* (1990) pp. 796–802
- 38.45 H. Yanco, L.A. Stein: *An Adaptive Communication Protocol for Cooperating Mobile Robots, From Animals to Animats 3. Proceedings of the Third International Conference on Simulation of Adaptive Behaviors* (MIT Press, Cambridge 1993) pp. 478–485
- 38.46 J.R. del Millán: *Learning Efficient Reactive Behavioral Sequences from Basic Reflexes in a Goal-Directed Autonomous Robot, From Animals to Animats 3. Proceedings Third International Conference on Simulation of Adaptive Behaviors* (MIT Press, Cambridge 1994) pp. 266–274
- 38.47 L. Parker: Learning in cooperative robot teams, *Proceedings International Joint Conference on Artificial Intelligence* (1993) pp. 12–23
- 38.48 M.J. Mataric: *Learning to Behave Socially, From Animals to Animats 3. Proceedings Third International Conference on Simulation of Adaptive Behaviors* (The MIT Press, Cambridge 1994) pp. 453–462
- 38.49 M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, K. Hosoda: Coordination of multiple behaviors acquired by a vision-based reinforcement learning, *Proceedings IEEE/RSJ/GI International Conference on Intelligent Robots and Systems*, Munich, Germany (1994)
- 38.50 J. McCarthy: Making robots conscious of their mental states, *AAAI Spring Symposium* (1995)
- 38.51 T. Smithers: *On why Better Robots Make it Harder, From Animals to Animats: Proceedings 3rd International Conference on Simulation of Adaptive Behavior* (MIT Press, Cambridge 1994) pp. 64–72
- 38.52 D. McFarland, T. Bösner: *Intelligent Behavior in Animals and Robots* (MIT Press, Bradford Book 1993)
- 38.53 P. Maes: *A Bottom-Up Mechanism for Behavior Selection in an Artificial Creature, From Animals to Animats. Proceedings First International Conference on Simulation of Adaptive Behavior* (MIT Press, Cambridge 1991) pp. 238–246
- 38.54 B.M. Blumberg, P.M. Todd, P. Maes: No bad dogs: Ethological lessons for learning in Hamsterdam, *From Animals to Animats: Proceedings International Conference on Simulation of Adaptive Behavior*, ed. by P. Maes, M.J. Mataric, J.-A. Meyer, J. Pollack, S.W. Wilson (1996) pp. 295–304
- 38.55 C. Breazeal, B. Scassellati: Infant-like social interactions between a robot and a human caregiver, *Adapt. Behav.* **8**(1), 49–74 (2000)
- 38.56 F. Michaud, M.T. Vu: Managing robot autonomy and interactivity using motives and visual communication, *Proceedings International Conference on Autonomous Agents* (1999) pp. 160–167

- 38.57 F. Michaud: EMIB – Computational architecture based on emotion and motivation for intentional selection and configuration of behaviour-producing modules, *Cogn. Sci. Q., Special Issue on Desires, Goals, Intentions and Values: Comput. Arch.* **3-4**, 340–361 (2002)
- 38.58 F. Michaud, P. Prijanian, J. Audet, D. Létourneau: *Artificial Emotion and Social Robotics, Distributed Autonomous Robotic Systems*, ed. by L.E. Parker, G. Bekey, J. Barhen (Springer, Berlin, Heidelberg 2000) pp. 121–130
- 38.59 A. Stoytchev, R. Arkin: Incorporating motivation in a hybrid robot architecture, *J. Adv. Comput. Intell. Intell. Inf.* **8**(3), 269–274 (2004)
- 38.60 S. Mahadevan, J. Connell: Automatic programming of behavior-based robots using reinforcement learning, *Artif. Intell.* **55**, 311–365 (1992)
- 38.61 M.J. Mataric: Reward functions for accelerated learning, *Proceedings 11th International Conference on Machine Learning*, New Brunswick, NJ, ed. by William W. Cohen, Haym Hirsh (Morgan Kaufman Publishers 1994) pp. 181–189
- 38.62 H. Gleitman: *Psychology* (NORTON, New York 1981)
- 38.63 M. Dorigo, M. Colombetti: *Robot Shaping: An Experiment in Behavior Engineering* (MIT Press, Cambridge 1997)
- 38.64 M. Nicolescu, M.J. Mataric: Learning and interacting in human–robot domains, *IEEE Transactions on Systems, Man, Cybernetics*, special issue on Socially Intelligent Agents – The Human in the Loop (2001)
- 38.65 A.K. McCallum: Hidden state and reinforcement learning with instance-based state identification, *IEEE Trans. Syst. Man Cybernet. – Part B: Cybernetics* **26**(3), 464–473 (1996)
- 38.66 F. Michaud, M.J. Mataric: Learning from history for behavior-based mobile robots in non-stationary environments, *Spec. Iss. Learn. Autonom. Robot. Mach. Learn./Autonom. Robot.* **31/5**, 141–167/335–354 (1998)
- 38.67 F. Michaud, M.J. Mataric: Representation of behavioral history for learning in nonstationary conditions, *Robot. Autonom. Syst.* **29**(2), 1–14 (1999)
- 38.68 A. Agha, G. Bekey: Phylogenetic and ontogenetic learning in a colony of interacting robots, *Autonom. Robot.* **4**(1), 85–100 (1997)
- 38.69 R.A. Brooks, L. Stein: Building brains for bodies, *Autonom. Robot.* **1**(1), 7–25 (1994)
- 38.70 B. Webb: *Robotic Experiments in Cricket Phonotaxis, From Animals to Animats 3. Proceedings Third International Conference on Simulation of Adaptive Behaviors* (MIT Press, Cambridge 1994) pp. 45–54
- 38.71 J.L. Jones: Robots at the tipping point, *IEEE Robot. Autom. Mag.* **13**(1), 76–78 (2006)
- 38.72 P. Rusu, E.M. Petriu, T.E. Whalen, A. Cornell, H.J.W. Spoelder: Behavior-based neuro-fuzzy controller for mobile robot navigation, *IEEE* **52**(4), 1335–1340 (2003)
- 38.73 R. Huq, G.K.I. Mann, R.G. Gosine: Behaviour modulation technique in mobile robotics using fuzzy discrete event system, *IEEE Trans. Robot.* **22**, 903–916 (2006)
- 38.74 L.E. Parker: Current research in multirobot systems, *Artif. Life Robot.* **7**(1–2), 1–5 (2003)
- 38.75 L.E. Parker, M. Chandra, F. Tang: *Enabling Autonomous Sensor-Sharing for Tightly-Coupled Cooperative Tasks, Multi-Robot Systems. From Swarms to Intelligent Automata*, Vol. III, ed. by L.E. Parker, F.E. Schneider, A.C. Schultz (Springer, Berlin, Heidelberg 2005) pp. 119–230
- 38.76 B.B. Werger, M.J. Mataric: Broadcast of Local Eligibility for Multi-Target Observation, *Proceedings of the 5th International Conference on Distributed Autonomous Robotic Systems* (2000) pp. 347–356
- 38.77 B.P. Gerkey, M.J. Mataric: Principled communication for dynamic multi-robot task allocation. In: *Experimental Robotics VII, LNCIS 271*, ed. by D. Rus, S. Singh (Springer, Berlin, Heidelberg 2001)
- 38.78 B.P. Gerkey, M.J. Mataric: Sold!: Auction methods for multi-robot coordination, *IEEE Trans. Robot. Autom.* **18**(5), 758–768 (2002)
- 38.79 B.P. Gerkey, M.J. Mataric: Pusher-watcher: An approach to fault-tolerant tightly-coupled robot coordination, *Proceedings IEEE International Conference on Robotics and Automation* (2002) pp. 464–469
- 38.80 L. Iocchi, D. Nardi, M. Piaggio, A. Sgorbissa: Distributed coordination in heterogeneous multi-robot systems, *Autonom. Robot.* **15**(2), 155–168 (2004)
- 38.81 M. Batalin, G. Sukhatme: Coverage, exploration and deployment by a mobile robot and communication network, *Telecommun. Syst.* **26**(2–4), 181–196 (2004)
- 38.82 A.W. Stroupe, T. Balch: Value-based action selection for observation with robot teams using probabilistic techniques, *Robot. Autonom. Syst.* **50**(2–3), 85–97 (2005), special issue on Multi-Robots in Dynamic Environments
- 38.83 R. Simmons, T. Smith, M.B. Dias, D. Goldberg, D. Hersherberger, A. Stentz, R. Zlot: A Layered Architecture for Coordination of Mobile Robots, *Proceedings from the NRL Workshop on Multi-Robot Systems Multi-Robot Systems: From Swarms to Intelligent Automata* (2002)
- 38.84 J. Nembrini, A. Winfield, C. Melhuish: *Minimalist Coherent Swarming of Wireless Networked Autonomous Mobile Robots, Proceedings of the 7th International Conference on Simulation of Adaptive Behavior* (MIT Press, Cambridge 2002) pp. 373–382
- 38.85 M. Egerstedt, X. Hu: Formation constrained multi-agent control, *IEEE Trans. Robot. Autom.* **17**(6), 947–951 (2001)
- 38.86 A. Olenderski, M. Nicolescu, S. Louis: A behavior-based architecture for realistic autonomous ship

- control, Proceedings, IEEE Symposium on Computational Intelligence and Games (2006)
- 38.87 M. Nicolescu, O.C. Jenkins, A. Olenderski: Learning behavior fusion estimation from demonstration, Proceedings, IEEE International Symposium on Robot and Human Interactive Communication (2006) pp. 340–345
- 38.88 K. Gold, B. Scassellati: Learning about the self and others through contingency, AAAI Spring Symposium on Developmental Robotics (2005)
- 38.89 M. Baker, H.A. Yanco: Automated street crossing for assistive robots, Proceedings of the International Conference on Rehabilitation Robotics (2005) pp. 187–192
- 38.90 M. Williamson: *Postural Primitives: Interactive Behavior for a Humanoid Robot Arm*, Proceedings of the International Conference on Simulation of Adaptive Behavior (MIT Press, Cambridge 1996)
- 38.91 M. Marjanovic, B. Scassellati, M. Williamson, R. Brooks, C. Breazeal: The Cog Project: Building a humanoid robot. In: *Computation for Metaphors, Analogy and Agents*, Vol. 1562 of Springer Lecture Notes in Artificial Intelligence, ed. by C. Nehaniv (Springer, Berlin, Heidelberg 1998)
- 38.92 A. Edsinger: Robot Manipulation in Human Environments. Ph.D. Thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science (2007)
- 38.93 C. Breazeal: Infant-like social interactions between a robot and a human caretaker, *Adapt. Behav.* **8**(1), 49–74 (2000)
- 38.94 H. Ishiguro, T. Kanda, K. Kimoto, T. Ishida: A robot architecture based on situated modules, Proceedings of the International Conference on Intelligent Robots and Systems (1999) pp. 1617–1623
- 38.95 T. Kanda, T. Hirano, D. Eaton, H. Ishiguro: Person identification and interaction of social robots by using wireless tags, Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (2003) pp. 1657–1664
- 38.96 F. Michaud, Y. Brosseau, C. Cote, D. Letourneau, P. Moisan, A. Ponchon, C. Raievsy, J.-M. Valin, E. Beaudry, F. Kabanza: Modularity and integration in the design of a socially interactive robot, Proceedings IEEE International Workshop on Robot and Human Interactive Communication (2005) pp. 172–177
- 38.97 F. Michaud, G. Lachiver, C.T. Le Dinh: Architectural methodology based on intentional configuration of behaviors, *Comput. Intell.* **17**(1), 132–156 (2001)
- 38.98 D. Letourneau, F. Michaud, J.-M. Valin: Autonomous robot that can read, *EURASIP J. Appl. Signal Process.* **17**, 1–14 (2004), Special Issue on Advances in Intelligent Vision Systems: Methods and Applications
- 38.99 J.-M. Valin, F. Michaud, B. Hadjou, J. Rouat: Localization of simultaneous moving sound sources for mobile robot using a frequency-domain steered beamformer approach, Proceedings IEEE International Conference on Robotics and Automation (2004) pp. 1033–1038
- 38.100 J.-M. Valin, F. Michaud, J. Rouat: Robust 3D localization and tracking of sound sources using beamforming and particle filtering, Proceedings International Conference on Audio, Speech and Signal Processing (2006) pp. 221–224
- 38.101 F. Michaud, C. Cote, D. Letourneau, Y. Brosseau, J.-M. Valin, E. Beaudry, C. Raievsy, A. Ponchon, P. Moisan, P. Lepage, Y. Morin, F. Gagnon, P. Giguere, M.-A. Roux, S. Caron, P. Frenette, F. Kabanza: Spartacus attending the 2005 AAAI Conference, Autonomous Robots, Special Issue on AAAI Mobile Robot Competition (2007)
- 38.102 F. Michaud, D. Letourneau, M. Frechette, E. Beaudry, F. Kabanza: Spartacus, scientific robot reporter, Proceedings of the Workshop on AAAI Mobile Robot Competition (2006)
- 38.103 E. Beaudry, Y. Brosseau, C. Cote, C. Raievsy, D. Letourneau, F. Kabanza, F. Michaud: Reactive planning in a motivated behavioral architecture, Proceedings American Association for Artificial Intelligence Conference (2005) pp. 1242–1247
- 38.104 K. Haigh, M. Veloso: Planning, execution and learning in a robotic agent, Proceedings Fourth International Conference on Artificial Intelligence Planning Systems (1998) pp. 120–127
- 38.105 S. Lemai, F. Ingrand: Interleaving temporal planning and execution in robotics domains, Proceedings National Conference on Artificial Intelligence (2004) pp. 617–622
- 38.106 F. Michaud, J.F. Laplante, H. Larouche, A. Duquette, S. Caron, D. Letourneau, P. Masson: Autonomous spherical mobile robotic to study child development, *IEEE Trans. Syst. Man. Cybernet.* **35**(4), 1–10 (2005)
- 38.107 F. Michaud, S. Caron: Roball, the rolling robot, *Autonom. Robot.* **12**(2), 211–222 (2002)
- 38.108 R.A. Brooks: *Cambrian Intelligence – The Early History of the New AI* (MIT Press, Cambridge 1999)
- 38.109 R. Pfeifer, C. Scheier: *Understanding Intelligence* (MIT Press, Cambridge 2001)
- 38.110 R.R. Murphy: *An Introduction to AI Robotics* (MIT Press, Cambridge 2000)
- 38.111 M.J. Mataric: *The Robotics Primer* (MIT Press, Cambridge 2007)
- 38.112 F. Martin: *Robotic Explorations: A Hands-On Introduction to Engineering* (Prentice Hall, Upper Saddle River 2001)
- 38.113 J.L. Jones, A.M. Flynn: *Mobile Robots – Inspiration to Implementation* (Peters, Wellesley 1993)

Distributed and Cellular Robots

Zack Butler, Alfred Rizzi

This chapter is organized according to a number of broad classes of problem where modular robotic systems can prove beneficial. For each such problem, the benefits of modularity are described, along with the ways that particular systems or proposed systems have explored those benefits. In particular, we discuss locomotion in Sect. 39.1, manipulation in Sect. 39.2, modular robot geometry in Sect. 39.3, and robust systems in Sect. 39.4. The systems under consideration in general have some level of independent computation on each module, and this discussion will focus on systems in which modules maintain some sort of kinematic constraint between them during operation. Compared to the types of multirobot teams described in Chap. 40, the systems of interest here are generally much more tightly coupled, both physically and conceptually. That is, we are primarily concerned with systems which, though they have many processors and independent actuators, have a single goal or small set of goals which can only be achieved collectively, rather than a set of goals

- 39.1 **Modularity for Locomotion** 911
 - 39.1.1 Self-Reconfigurable Robot Locomotion 912
 - 39.1.2 Physically Cooperative Mobile Robots..... 914
- 39.2 **Modularity for Manipulation**..... 914
 - 39.2.1 Independent Manipulators..... 914
 - 39.2.2 Reconfigurable Manipulators 915
- 39.3 **Modularity for Geometric Reconfiguration of Robot Systems** 915
 - 39.3.1 Manually Reconfigured Systems 916
 - 39.3.2 Shape Generation via Self-Reconfiguring Systems 916
 - 39.3.3 Configuration Optimization 917
 - 39.3.4 Self-Replicating Systems 918
- 39.4 **Modularity for Robustness**..... 918
- 39.5 **Conclusions and Further Reading**..... 918
- References** 919

which can be apportioned to single (or a small number of) robots within the team.

For many families of tasks in robotics, including manipulation and locomotion, different instances of the task can be best solved by robots with different kinematics. For example, both legged robots and wheeled robots can be effective for locomotion, but each type is better suited to a particular type of environment (rough or generally smooth). Likewise, manipulator arms with different geometry can achieve good ma-

nipulability over different workspaces. The use of modularity in robots, whether by developing physically interchangeable parts for a single robot or creating completely independent robots that can join together as the situation demands, can make it possible for a single overall robotic system to achieve a variety of kinematic configurations that a fixed architecture can not.

39.1 Modularity for Locomotion

Modular robots can achieve a very wide variety of forms for locomotion over (or even through) rough terrain. A single robot system may be able to surmount obstacles

many times the size of the modules itself, while also being able to navigate tunnels just larger than a single module. Some systems have been designed primarily

for multimodal locomotion, while other more general-purpose machines can also perform in this way, albeit less efficiently.

39.1.1 Self-Reconfigurable Robot Locomotion

Self-reconfigurable robots are a potentially very powerful type of system in which the modules move themselves with respect to each other to autonomously attain different configurations for different purposes. Systems constructed to date generally fall into two categories, *chain based* and *lattice based*, defined by how a collection of modules lies in space. Chain-based systems form essentially one-dimensional skeleton structures which can move in a continuous way, while lattice-based structures tend to have space-filling modules that form configurations that are relatively dense in two or three dimensions.

In chain-based systems, the modules generally connect as links in a chain, such that most modules connect to two other modules along an axis about which the module can rotate. Additional junction modules provide for more complex structures, such as legged spiders in addition to wheels and snakes. Some examples of such systems are the PolyBot [39.1] and the configurable robot (CONRO) system [39.4]. There are two major challenges when building chain-based systems. First of all, control of reconfiguration presents an interesting kinematic problem – turning a snake into a loop requires coordination of all the joints to bring the two ends of the snake close enough to allow physical connection. This is similar to traditional inverse kinematics with computation distributed over the joints of the system.

Once a chain-based system has achieved a given configuration, in order to progress over the terrain, the

modules must produce a successful periodic gait. A gait is usually specified similarly to legged systems, that is, each joint has a schedule of angle with respect to time. The challenge here is that the gait will depend on the configuration – snakes of different lengths may require different gaits, and legged constructions with a varying number of legs are also possible. The work of Zhang et al. [39.5] on the Polybot system describes a set of scalable configuration types (loops, legs, snakes) and automated transitions between them as well as automatic gait generators. This work uses phase automata to generate the periodic angle for each joint, but specified in a way that can be easily recomputed as the configuration changes.

Modules in lattice-based systems fill space much like building blocks. This allows these robots to construct a much wider variety of shapes, and as such they may be able to surmount larger obstacles, but at the cost of less efficient locomotion. For example, as a wheeled or legged construct, a chain-based system cannot climb obstacles significantly higher than traditional wheeled or legged robots can, whereas a lattice-based robot can make scaffolding-like shapes limited only by the number of modules available, with modules then climbing the scaffolding and reforming a group on the opposite side of the obstacle. Here the hardware specifics become very important, as these systems perform locomotion by reconfiguring, that is, the modules will move over the remainder of the structure one at a time for the group to move forward. Figure 39.2 shows an example of lattice-based modules that actuate through compression and expansion climbing stairs using a hard-coded motion sequence.

Locomotion via shape change was formalized in a cellular-automata-like framework in [39.6] – in this work, each module independently runs a set of geometric rules and depending on the state of its neighborhood

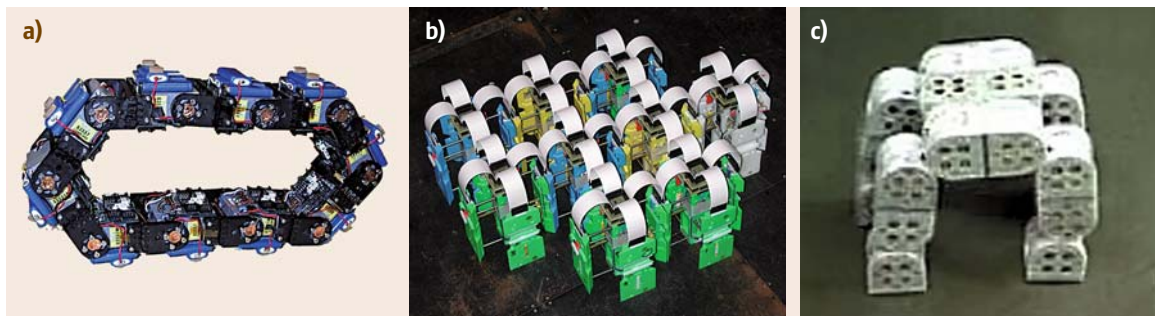


Fig. 39.1a–c Examples of self-reconfigurable robots: (a) the Polybot, a chain-based system [39.1]; (b) the Crystal, a two-dimensional lattice-based system [39.2]; (c) MTRAN [39.3], which can take on the characteristics of either type

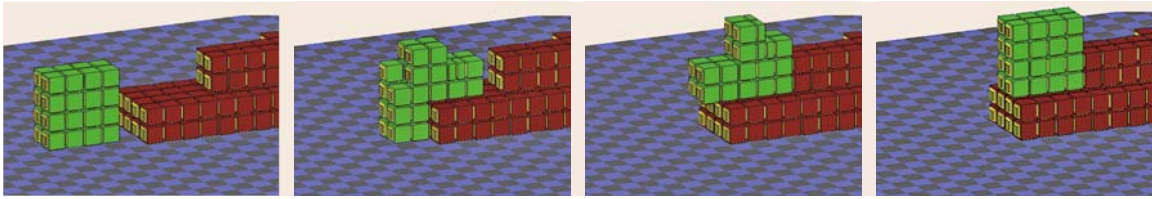


Fig. 39.2 A self-reconfigurable robot made of compressible modules climbing stairs that are two modules tall

(where other modules and obstacles are located), may move in a given direction. This technique was shown to ensure progress and avoid deadlock for straight-line motion over uneven terrain, traversing cliffs many modules high as well as narrow tunnels, and turning on flat ground, but is not trivially generalizable to more complex operations.

Lattice-based robots also have a wider variety of module types. Some systems are purely planar (e.g., [39.9, 10]) while others are three dimensional (e.g., [39.11]). They can be simple cubes or more complex shapes, though in general a single module is incapable of significant motion in isolation, instead requiring a substrate of other modules to connect to and walk across. Connections between modules can be mechanical [39.12] or magnetic [39.9]. The work of [39.6] proposed a generic model of a cubic module that can translate over other modules and make convex transitions, and showed that several different hardware systems can implement this model (though some require multiple hardware modules, termed metamodules, to implement a single generic module).

Recently this divide has been bridged with hardware than can operate in both chain-based and lattice-based modes. Modular transformer (**MTRAN**) and its successors [39.13] consist of 2-DOF modules, with each module containing two semicylindrical parts that rotate about a link between them. Each half of the module can make three connections to other modules through magnetic connectors that require energy only to disconnect. This robot can also operate completely untethered, including connecting and disconnecting, to effect true self-reconfiguration. This system has demonstrated self-reconfiguration in a lattice-based mode (where the discrete nature of the lattice makes the kinematics and therefore the docking problem much simpler) followed by conversion to a four-legged walker which uses the joints of the modules in a continuous fashion. The Superbot [39.14] is another recently developed system which promises to be at least as functional as the **MTRAN** due to an additional degree of freedom per module, however at present the ability to physically connect

and disconnect autonomously is still under development.

In all of these systems, constructing functional and robust hardware is a critical challenge. The primary con-

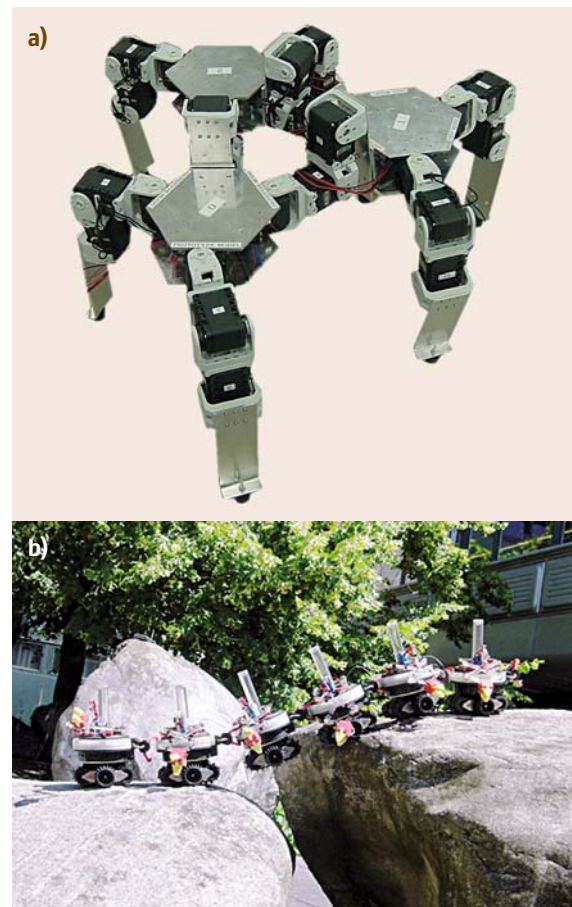


Fig. 39.3a,b Examples of independently capable mobile robots which can join together to perform more difficult tasks. (a) Three modules of the Ashigaru system [39.7] combined into a six-legged walker. (b) Swarmbots [39.8] connected together traversing a gap larger than a module in width

sideration is that of the connection mechanism – these will require fairly rigid connection, often including multiple electrical connections for communication and/or power transfer, while also allowing for physical connection in the face of position error. The position error can either be handled through sensing, such as the multiple beacons used in the CONRO modules, or through mechanical means, such as the grippers used in the Molecule robot. Connectors are especially critical for lattice-based systems, which must perform connection and disconnection to effect any significant geometric change.

39.1.2 Physically Cooperative Mobile Robots

In other situations, the locomotion requirements may be largely straightforward, with only certain specific areas requiring unusual abilities. In these cases, it may be more appropriate to use more traditional mobile robots as modules that can travel efficiently on their own but can also make rigid or semirigid connection to achieve greater capability.

One example of this is the Gunryu system proposed by Hirose et al. [39.15]. In this system, each module is a stand-alone tracked mobile robot approximately 0.5 m in length with a two-joint arm mounted on it. Each module can autonomously navigate moderately difficult terrain. Upon encountering very steep terrain or a hole in the ground larger than one module, the module can

use its manipulator arm to hold onto another module. The authors were able to demonstrate that two modules connected together could traverse steeper ground than a single module. Key to this ability is the strength of the manipulator; it must be sufficiently strong to hold the weight of another module. A more recent robot called Ashigaru uses modules that each have three legs and can execute simple crawling gaits alone. Modules can grab each other in a variety of ways, as each leg can act as a grabber. Demonstrations have included forming quadrupeds and hexapods (such as in Fig. 39.3a), climbing steps taller than single modules are able to, and even using one module as a simple manipulator while it is held by other modules. It is currently manually reconfigured, though plans include the ability for self-reconfiguration [39.7].

More extensive work in this line is being carried out by in the SWARM-BOTS project [39.16]. The modules in this system are significantly smaller (circular robots 120 mm in diameter) but have a similar gripper that can grab other modules to allow for cooperative locomotion. In this case, the gripper is a 2-DOF wrist rather than a serial manipulator, which allows for greater rigidity. These modules have demonstrated the ability to cross a gap greater than one module diameter wide, as in Fig. 39.3b [39.8]. Each module also has a short serial manipulator to perform a wider variety of manipulation tasks, and the overall system has also demonstrated simple forms of cooperative manipulation.

39.2 Modularity for Manipulation

Modularity can also be a great advantage when manipulating a wide variety of payloads. In particular, the underlying kinematics of a manipulation system may be changed rapidly (either online or offline, depending on the system) so that the workspace and manipulability are most appropriate for a given task.

39.2.1 Independent Manipulators

One common task for which a group of robots can be beneficial is that of pushing or carrying a large object. If the object is significantly larger than a single robot can handle, a group is the only way to move it through the environment. When implementing such a system, the key issues include the frequency of data transfer and the effective sharing of the load. In order for the robots to maintain control over the object, the forces inherent must be shared so that no single robot is overwhelmed.

Khatib et al. proposed the virtual linkage as an solution to this problem [39.17].

Force control is less of a concern in a quasistatic pushing problem – in this case, the challenge is in producing the appropriate trajectory for the pushed object given the limited range of forces that can be applied by each pusher. Synchronization is still important in such systems, however. Mataric et al. [39.18] demonstrated cooperative pushing with two legged robots moving a box that neither could move alone, and showed that in general, without synchronization, one would move faster than the other and misorient the box or move past the end of the box. In this work, the synchronization is done through explicit communication and time-slicing of control, that is, each robot has a time in which it can move, after which it passes control (and sensory information) to its colleague. This explicit synchronization is not always necessary, however – if the robots have suffi-

cient sensing and understanding of the mechanics of the pushing task, they can implicitly cooperate through the pushing task itself. That is, as the pushed object moves, each robot can infer the state of the overall system and choose its actions and role appropriately. This idea of information in the task mechanics was developed by Donald et al. [39.19], where they provide a set of transformations for information channels between the virtual and the physical.

Cooperative manipulation can also be achieved by modules that are not necessarily mobile. Some manufacturing-oriented systems have been proposed in which the manipulation modules can be much simpler than otherwise would be required because many modules can work together to perform the required tasks. One such system is the virtual vehicle of Luntz et al. [39.20]. In this system, each module is simply a wheel which can be rotated. By assembling a large number of them in a bounded region, a programmable conveyor is created. Multiple objects can be handled simultaneously at different points in the system, and can be handled differently (e.g., one being stopped and rotated in place while another is moved past). Again, synchronization is critical here – the motion of an object is determined by the total force and torque acting on it, which in turn is determined not only by the velocity and angle of each wheel under it, but also the relative location of each wheel. In order for the object to maintain the correct trajectory, all wheels must inject the appropriate force for their location, which can change as the object moves. One way to handle this complexity is to simplify the task such that the forces are not time dependent, for example, simply operating as a conveyor belt in which all wheels move the same direction, or having two portions of the system push toward each other to move objects to a given location.

Another system in which the use of multiple modules allows for the use of simpler robots is the minifactory of Hollis et al. [39.21]. Here, traditional 4-DOF assembly operations are performed by two robots, each with two DOF. One robot is an overhead two-axis ($Z-\theta$) manipulator and the other an $X-Y$ planar motor. The simplification of the robots, in particular the elimination of a serial chain in the overhead manipulator, allows for greatly increased precision and the performance of

micron-scale automated assembly. Here, however, the goal is to perform force-guided assembly, requiring the closing of a 1 kHz control loop between the two independent robots.

39.2.2 Reconfigurable Manipulators

A very different type of modularity for manipulation is that for a more or less traditional robot arm that can be easily rebuilt or reconfigured to achieve different kinematics. If the arm is constructed as a sequence of identical modular components (joints), these can then be attached to provide as many degrees of freedom as needed, with the appropriate workspace. The reconfigurable modular manipulator system (RMMS) was designed with these aims in mind [39.22]. The challenge here is to calculate the inverse kinematics and the dynamics automatically as the system changes, and in fact the RMMS does provide this functionality, though these systems usually have centralized computation with at most local joint controllers. This type of simplicity through modularity can be taken to a more extreme point, in which each module is a linkage with binary actuation, that is, only two possible states. Combining many of these binary modules with different geometry in different configurations creates robots with different workspaces from the same selection of components [39.23]. Here the kinematics and inverse kinematics are discrete, though the inverse kinematics are not closed form since the workspace is a set of discrete points in space. However, the control scheme is much simpler, since little or no feedback is required to direct the endpoint to a given location once the desired linkage positions are determined.

One of the earliest systems described in the field of self-reconfiguration is the cellular robot (CEBOT) of Fukuda et al. [39.24]. This system falls between the extremes of pure self-reconfiguring modules and mobile robots, as it does include modules which are independently mobile, but also includes nonmobile modules and expects that several modules will be used for most tasks. In one task described for CEBOT, individual modules can move through a small opening into a tank, and then reconfigure into a larger manipulator to perform operations across the entire tank.

39.3 Modularity for Geometric Reconfiguration of Robot Systems

The use of modularity allows for systems of many independent components to be reconfigured quickly between

task operations with a minimum of effort. This is useful in manufacturing systems, where the modules may

be capable of stand-alone operation, as well as self-reconfiguring robots, where sensors can be repositioned as the robot performs its tasks. In both cases, there are several issues to consider: ease of reconfiguration, determination of the current configuration, choosing good configurations for a particular task, and the design of the underlying task-performance algorithms for use with arbitrary configurations.

39.3.1 Manually Reconfigured Systems

Traditional manufacturing can be performed in workcells, in which usually a single robot has access to several stations and performs a series of assembly operations. These workcells are designed to be fairly modular from a component point of view, swapping different portions of the product in and out as the manufacturing task changes. Several workcells can then be laid out along a conveyor or other line mechanism to create an assembly line. While they may be very similar in construction, they tend to be fairly independent in operation. Scheduling in the traditional sense may be important, but real-time interactions are less likely to be.

In the Minifactory project, on the other hand, the entire system including the conveyances is part of a single modular system. All modules conform to a common interface, both in hardware and software [39.21]. This allows rapid reconfiguration, with each robot able to be placed in the system with clamps and simple connectors. The robots share a common high-level language and a centralized simulation environment, so that tasks can be programmed into the system ahead of time. When the modules are assembled, the robots that ferry the products through the system can also be used to discover the exact locations of all of the components as placed. The task algorithms, written under a common framework, are then automatically updated with the exact locations

to be visited. This allows for a smooth and efficient transition from a simulated assembly line to an operating set of hardware.

39.3.2 Shape Generation via Self-Reconfiguring Systems

Self-reconfigurable robots, as the name implies, can autonomously change their overall shape to fit the task at hand. In addition to using this capability to perform a variety of locomotion tasks, lattice-based systems in particular have the capability to make nearly arbitrary three-dimensional shapes. This may be useful for generating different sizes and shapes of payload carriers, manipulators, or other devices. In more recent research, systems of heterogeneous modules, where the capabilities of each module are not necessarily the same, have again received more attention [39.25]. These may be useful for positioning cameras or other specialized sensors at varying positions during the completion of a task even as the shape of the system remains the same. Figure 39.4 shows a simulation of a heterogeneous reconfiguration from a chair to a table – in this technique, the system first achieves the desired shape without regard to module type (denoted by module color), after which the modules physically sort themselves into the correct locations.

To perform self-reconfiguration, the modules in a system need to cooperate to plan their motions so that they can form the desired configuration. In common practice, the modules run without a central controller, but the global desired shape in some specification is given to all modules, at which point each module makes local decisions about if and where to move. To perform the reconfiguration, primarily local algorithms as well as more global planners have been developed.

Local shape-formation techniques tend to be connection based, that is, the desired shape is specified by

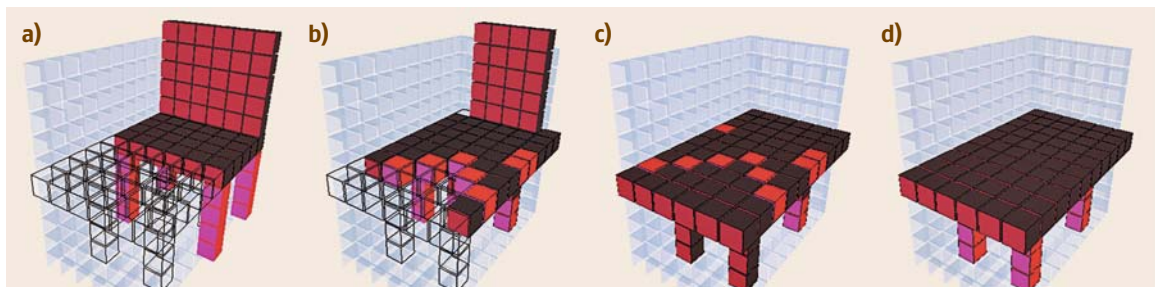


Fig. 39.4a–d Heterogeneous reconfiguration of a chair-shaped system to a table shape. (a) Initial configuration. (b) Modules moving in parallel to achieve the correct configuration. (c) Final system shape. (d) Final module locations based on heterogeneous module types

listing the connections to be made between modules. Each module is then responsible for ensuring it has the appropriate set of modules connected to it. *Tomita* et al. first proposed a shape generation scheme of this nature for two-dimensional systems [39.26], in which a single seed module collected modules around in successive layers, with each layer collecting appropriate modules as specified by the global design. Along a similar vein, work by *Lipson* et al. [39.27] constructs shapes out of modules that do not contain traditional actuation, but rather exist in a moving fluid medium. The modules can only control when and where to allow other modules to connect to them to build the desired shape, and so a connection-based scheme is natural. Recent work by *Klavins* [39.28] avoids the need for a seed module, and uses reaction network theory to allow modules that move stochastically (e.g., due to external forces) to intelligently detach and reattach and achieve many copies of the desired shape in an efficient and reliable way.

Global techniques tend to look more like traditional path planners – the modules discover local mismatches between the current configuration and the desired configuration, and paths are planned in a distributed fashion for modules to move into empty locations. One interesting feature is that the available paths are over the surface of other modules, so there is computation available wherever the search needs to take place, and distributed planning is quite natural. In addition, the paths are naturally discrete, so no artificial constraints need to be applied to perform the planning.

Here there are two main concerns: first, a matching problem must be solved, to ensure that each goal location is the destination of only one module and that each location is accounted for; and second, deadlock must be avoided and traversability maintained if and when multiple plans are being generated and executed within the system. The first can be solved by having each goal location spoken for by a particular module, and having that module initiate a path search for a module that can fill the location. In addition, these techniques also tend to build the goal shape one layer at a time, starting from the current shape and extending outward. Each individual path search must be filled by only one module, either by executing a depth-first search that terminates when a capable module is found (which is the approach taken in [39.29]) or having some other way to choose a module if multiple modules attempt to fill the requested location. Modules also must know to attempt motion only if they are not required at their current position and can move without disconnecting the overall structure. *MeltGrow* [39.30] and its heterogeneous successor,

MeltSortGrow, handle this latter problem by moving all the modules to an intermediate shape (a line) from which the end of the line can be locally detected and only this module will move to a goal location. However, modules can detect whether they can will disconnect the structure through a depth-first search from their neighbors, and so this melting step is unnecessary. During motion, care must be taken to ensure that paths are still valid as other modules begin to move. This can be done through serialization or through a localized traffic control scheme [39.29], in which modules communicate to take turns passing through a particular area. Shape reconfiguration algorithms can be shown to require $O(n^2)$ module motions for n modules [39.31], but usually only $O(n)$ time if the motions are executed in parallel.

If the systems are heterogeneous, that is, the modules are of different types, modules must also determine which type is required at a given location, and only modules of that type can respond to the particular request. *Fitch* et al. [39.25] propose a planning system for identically sized but arbitrarily typed modules in which the correct shape is first achieved regardless of type, after which modules swap location. This algorithm still only requires $O(n^2)$ motion under most circumstances and can be efficiently executed in a distributed way.

39.3.3 Configuration Optimization

The determination of an appropriate configuration is another key issue in both manually reconfigured and self-reconfiguring systems, but has been little discussed in the literature. One common approach is an empirical one – to simulate a particular task or set of tasks (e.g., an assembly process) under different configurations and compare efficiency in each case. For example, in describing the *RMMS* system, [39.22] mentions that a simulator is available for determining the best configuration for a particular manipulation task. Similar work was performed for the *CEBOT* system, in which the location of the manipulator as well as the required number of modules and their configuration was optimized [39.32]. There is work with binary manipulators to find an optimal configuration for a given set of points to be reached, in which Lagrange multipliers are used to minimize the change from a base structure to a structure that can achieve the desired locations [39.23].

In systems such as self-reconfiguring robots, where hundreds of modules may be placed in fairly arbitrary ways, there are vast numbers of configurations to be considered, and the tasks themselves can be more complex. An evolutionary approach to optimizing con-

figurations of modular (though not self-reconfiguring) robots is described in Chap. 61. More general configuration optimization approaches from multirobot teams may be applicable for tasks such as surveillance, though they would need to be modified to handle the added capabilities of self-reconfiguring systems.

39.3.4 Self-Replicating Systems

One type of system that obviates the question of configuration determination is the self-replicating mod-

ular robot. In these systems, one group of modules builds a copy from a supply of modules. While the theory of self-replication was established by *von Neumann* [39.33], engineering such a robotic system is a significant challenge. This capability has been demonstrated in restricted form by *Suthakorn et al.* [39.34], in which the robots are constructed of four fairly complex modules, and one robot can construct a second, and by *Zykov et al.* [39.35] in a system of more traditional lattice-based self-reconfiguring modules.

39.4 Modularity for Robustness

Finally, the use of a large number of simple modules to complete a task allows for robustness in the event of failure. If the modules are completely homogeneous, then they can simply be swapped out and replaced. If the system is engineered with sufficient redundancy, modules may even be done without as they fail. This process of self-repair may be possible to execute in an autonomous fashion, as has been proposed for some self-reconfigurable systems [39.26, 36]. In these cases, the failed module is ejected from the system either by selective disassembly to get the failed module to the surface, or by a pushing gait that moves the module. The DRAGON connector [39.37] can always be detached from either side in the event of module failure, so that the system can continue with minimal disruption. How-

ever, despite the presumed importance of this self-repair capability of modular systems, this has not been a major focus of research.

On the other hand, systems which are manually reconfigured may not have the option to evict a nonfunctional module. Instead, they may be built with inherent redundancy so that nonfunctional modules can simply be handled, either actively or passively. For example, in the virtual vehicle, the failure of a single unit may not affect the overall performance if the objects in transport are large relative to the actuators. Likewise, redundant manipulators such as the RMMS [39.22] can continue to function if a single joint seizes, though this must be actively detected so that their changed kinematics and dynamics can be taken into account.

39.5 Conclusions and Further Reading

The benefits of modularity can be seen in many different task domains, whether the systems change their configuration on their own or require assistance from humans. The modularity can be used to simply create larger machines, for example, for more capable locomotion, or more complex machines that still retain a notion of simplicity at the module level. At present, hardware systems have largely remained in the research realm, while algorithmic progress has been notably more advanced in several areas, such as self-reconfiguration. It is hoped that the costs associated with modular robot development can be reduced as a byproduct of mass production,

enabling more deployments that take advantage of the robustness and versatility inherent to such systems.

Interested readers may find more details of some self-reconfiguring robots in a survey paper [39.38], and workshops in this area are often held in conjunction with international conferences, such as the 2006 Robotics: Science and Systems conference, and the 2007 International Conference on Intelligent Robots and Systems. Distributed robotic systems in general are the subject of the biennial distributed autonomous robotics systems (DARS) conference, including many papers relevant to this topic.

References

- 39.1 M. Yim, D. Duff, K. Roufas: PolyBot: A modular reconfigurable robot, Proc. of IEEE ICRA (2000) pp. 514–520
- 39.2 Z. Butler, D. Rus: Distributed motion planning for modular robots with unit-compressible modules, Int. J. Robot. Res. **22**(9), 699–716 (2003)
- 39.3 S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura, S. Kokaji: Hardware design of modular robotic system, Proc. of IROS (2000) pp. 2210–2217
- 39.4 A. Castano, W.-M. Shen, P. Will: CONRO: Towards deployable robots with inter-robots metamorphic capabilities, Auton. Robot. **8**(3), 309–324 (2000)
- 39.5 Y. Zhang, M. Yim, C. Eldershaw, D. Duff, K. Roufas: Scalable and reconfigurable configurations and locomotion gaits for chain-type modular reconfigurable robots, Proceedings of the International Symposium on Computational Intelligence in Robotics and Automation (2003) pp. 893–899
- 39.6 Z. Butler, K. Kotay, D. Rus, K. Tomita: Generic decentralized locomotion control for lattice-based self-reconfigurable robots, Int. J. Robot. Res. **23**(9), 919–938 (2004)
- 39.7 M. Ohira, R. Chatterjee, T. Kamegawa, F. Matsuno: Development of three-legged modular robots and demonstration of collaborative task execution, Proc. of IEEE Int'l. Conf. on Robotics and Automation (2007) pp. 3895–3900
- 39.8 V. Trianni, S. Nolfi, M. Dorigo: Cooperative hole avoidance in a swarm-bot, Robot. Auton. Syst. **54**(2), 97–103 (2006)
- 39.9 S. Murata, H. Kurokawa, S. Kokaji: Self-assembling machine, Proc. of IEEE ICRA (1994) pp. 442–448
- 39.10 D. Rus, M. Vona: A physical implementation of the crystalline robot, Proc. of IEEE ICRA (2000)
- 39.11 C. Ünsal, P. Khosla: Mechatronic design of a modular self-reconfiguring robotic system, Proc. of IEEE ICRA (2000) pp. 1742–1747
- 39.12 K. Kotay, D. Rus, M. Vona, C. McGray: The self-reconfiguring robotic molecule: design and control algorithms, Proc. of the Workshop on Algorithmic Foundations of Robotics (1998)
- 39.13 S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, S. Kokaji: M-TRAN: Self-reconfigurable modular robotic system, IEEE/ASME Trans. Mechatron. **7**(4), 431–441 (2002)
- 39.14 W.-M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, J. Venkatesh: Multimode locomotion for reconfigurable robots, Auton. Robot. **20**(2), 165–177 (2006)
- 39.15 S. Hirose, T. Shiratsu, F.E. Fukushima: A proposal for cooperative robot “gunryu” composed of autonomous segments, Proceedings of the International Conference on Intelligent Robots and Systems (1994) pp. 1532–1538
- 39.16 F. Mondada, L.M. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneubourg, M. Dorigo: The cooperation of swarm-bots, IEEE Robot. Autom. Mag. **12**(2), 21–28 (2005)
- 39.17 O. Khatib, K. Yokoi, K. Chang, D. Ruspin, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robot. Syst. **13**(11), 755–764 (1996)
- 39.18 M. Mataric, M. Nilsson, K. Simsarian: Cooperative multi-robot box pushing, Proceedings of the International Conference on Robotics and Automation (1995) pp. 556–561
- 39.19 B. Donald, J. Jennings, D. Rus: Information invariants for distributed manipulation, Int. J. Robot. Res. **16**(5), 673–702 (1997)
- 39.20 J. Luntz, W. Messner, H. Choset: Closed-loop operation of actuator arrays, Proc. of IEEE Int'l. Conf. on Robotics and Automation (2000) pp. 3666–3671
- 39.21 A.A. Rizzi, J. Gowdy, R.L. Hollis: Agile assembly architecture: An agent-based approach to modular precision assembly systems, Proc. of IEEE Int'l. Conf. on Robotics and Automation (1997) pp. 1511–1516
- 39.22 C. Paredis, H.B. Brown, P. Khosla: A rapidly deployable manipulator system, Proceedings of the International Conference on Robotics and Automation (1996) pp. 1434–1439
- 39.23 G.S. Chirikjian: A binary paradigm for robotic manipulators, Proc. of IEEE Int'l. Conf. on Robotics and Automation (1994) pp. 3063–3069
- 39.24 T. Fukuda, T. Ueyama: *Cellular Robotics and Micro-robotic Systems* (World Scientific, Singapore 1994)
- 39.25 R. Fitch, Z. Butler, D. Rus: In-place distributed heterogeneous reconfiguration planning, Proc. of Distributed Autonomous Robotic Systems (2004)
- 39.26 K. Tomita, S. Murata, H. Kurokawa, E. Yoshida, S. Kokaji: Self-assembly and self-repair method for a distributed mechanical system, IEEE Trans. Robot. Autom. **15**(6), 1035–1045 (1999)
- 39.27 P. White, V. Zykov, J. Bongard, H. Lipson: Three dimensional stochastic reconfiguration of modular robots, Proceedings of Robotics: Science and Systems (Cambridge 2005)
- 39.28 E. Klavins: Tuning reaction networks for programmed self-organization, Proceedings of the Third Conference on the Foundations of Nanoscience (2006) pp. 34–37
- 39.29 S. Vassilvitskii, M. Yim, J. Suh: A complete, local and parallel reconfiguration algorithm for cube style modular robots, Proc. of IEEE ICRA (2002) pp. 117–122
- 39.30 M. Vona, D. Rus: Self-reconfiguration planning with compressible unit modules, Proc. of IEEE Int'l. Conf. on Robotics and Automation (1999) pp. 2513–2520

- 39.31 G. Chirikjian, A. Pamecha: Bounds for self-reconfiguration of metamorphic robots, Proceedings of the International Conference on Robotics and Automation (1996) pp.1452–1457
- 39.32 T. Fukuda, S. Nakagawa, Y. Kawauchi, M. Buss: Structure decision method for self organizing robot based on cell-structure robot, Proceedings of IEEE International Conference on Robotics and Automation (1989) pp. 695–700
- 39.33 J. von Neumann: *Theory of Self-Replicating Automata* (Univ. of Illinois Press, Chicago 1966)
- 39.34 J. Suthakorn, Y.T. Kwon, G. Chirikjian: An autonomous self-replicating robotic system, Proc. of the International Conference on Advanced Intelligent Mechatronics (2003) pp.137–142
- 39.35 V. Zykov, E. Mytilinaios, B. Adams, H. Lipson: Self-reproducing machines, *Nature* **435**, 163–164 (2005)
- 39.36 R. Fitch, D. Rus, M. Vona: A basis for self-repair robots using self-reconfiguring crystal modules, *Intelligent Autonomous Systems 6* (2000)
- 39.37 M. Nilsson: Connectors for self-reconfiguring robots, *Trans. Mechatron.* **7**(4), 473–474 (2002)
- 39.38 D. Rus, Z. Butler, K. Kotay, M. Vona: Self-reconfiguring robots, *Commun. ACM* **45**(3), 39–45 (2002)

40. Multiple Mobile Robot Systems

Lynne E. Parker

Within the context of multiple *mobile* robot systems, this chapter explores the current state of the art. After a brief introduction, we first examine architectures for multirobot cooperation, exploring the alternative approaches that have been developed. Next, we explore communications issues and their impact on multirobot teams in Sect. 40.3, followed by a discussion of swarm robot systems in Sect. 40.4. While swarm systems typically assume large numbers of homogeneous robots, other types of multirobot systems include heterogeneous robots. We therefore next discuss heterogeneity in cooperative robot teams in Sect. 40.5. Once robot teams allow for individual heterogeneity, issues of task allocation become important; Sect. 40.6 therefore discusses common approaches to task allocation. Section 40.7 discusses the challenges of multirobot learning, and some representative approaches. We outline some of the typical application domains which serve as test beds for multirobot systems research in Sect. 40.8. Finally, we conclude in Sect. 40.9 with some summary remarks and suggestions for further reading.

40.1 History	922
40.2 Architectures for Multirobot Systems	922
40.2.1 The Nerd Herd	923
40.2.2 The ALLIANCE Architecture	923
40.2.3 The Distributed Robot Architecture	924
40.3 Communication	925
40.4 Swarm Robots	926
40.5 Heterogeneity	928
40.6 Task Allocation	930
40.6.1 Taxonomy for Task Allocation	930
40.6.2 Representative Approaches	930
40.7 Learning	932
40.8 Applications	933
40.8.1 Foraging and Coverage	933
40.8.2 Flocking and Formations	933
40.8.3 Box Pushing and Cooperative Manipulation	934
40.8.4 Multitarget Observation	934
40.8.5 Traffic Control and Multirobot Path Planning	934
40.8.6 Soccer	935
40.9 Conclusions and Further Reading	935
References	936

Researchers generally agree that multirobot systems have several advantages over single-robot systems [40.1, 2]. The most common motivations for developing multirobot system solutions are that:

1. the task complexity is too high for a single robot to accomplish;
2. the task is inherently distributed;
3. building several resource-bounded robots is much easier than having a single powerful robot;
4. multiple robots can solve problems faster using parallelism; and

5. the introduction of multiple robots increases robustness through redundancy.

The issues that must be addressed in developing multirobot solutions are dependent upon the task requirements and the sensory and effector capabilities of the available robots.

The types of robots considered in the study of multiple *mobile* robot systems are those robots that move around in the environment, such as ground vehicles, aerial vehicles, or underwater vehicles. This chapter focuses specifically on the interaction of multiple *mobile*

robots, as distinguished from other types of multirobot interaction. For example, a special case of multiple mobile robot systems are the *reconfigurable* or *modular* robots that interconnect with each other for the purposes of navigation or manipulation. This type of multirobot system is covered in detail in Chap. 39. *Networked* robotics, covered in Chap. 41, is also very closely re-

lated to multiple mobile robot systems; however, the focus in networked robotics is on systems of robots, sensors, embedded computers, and human users that are all connected by networked communication. Another variant of multirobot cooperation is multiple manipulator arm cooperation; Chap. 29 describes these systems in detail.

40.1 History

Since the earliest work on multiple mobile robot systems in the 1980s, the field has grown significantly, and covers a large body of research. At the most general level, approaches to multiple mobile robot systems fall into one of two broad categories: *collective swarm* systems and *intentionally cooperative* systems. *Collective swarm* systems are those in which robots execute their own tasks with only minimal need for knowledge about other robot team members. These systems are typified by the assumption of a large number of homogeneous mobile robots, in which robots make use of local control laws to generate globally coherent team behaviors, with little explicit communication among robots. On the other hand, robots in *intentionally cooperative* systems have knowledge of the presence of other robots in the environment and act together based on the state, actions, or capabilities of their teammates in order to accomplish the same goal. Intentionally cooperative systems vary in the extent to which robots take into account the actions or state of other robots, and can lead to either strongly or weakly cooperative solutions [40.3]. *Strongly cooperative* solutions require robots to act in concert to achieve the goal, executing tasks that are not trivially serializable. Typically, these approaches require some type of communication and synchronization among the robots. *Weakly cooperative* solutions allow robots to have periods of operational independence, subsequent to coordinating their selection of tasks or roles. Inten-

tionally cooperative multirobot systems can deal with heterogeneity in the robot team members, in which team members vary in their sensor and effector capabilities. In these teams, the coordination of robots can be very different from in collective swarm approaches, since robots are no longer interchangeable.

Most of the work specific to multiple mobile robot cooperation can be categorized into a set of key topics of study. These topics, which are the foci of this chapter, include *architectures*, *communication*, *swarm robots*, *heterogeneity*, *task allocation*, and *learning*. *Architectures* and *communication* in multirobot systems are relevant for all types of multirobot systems, as these approaches specify how the robot team members are organized and interact. *Swarm robots* is a particular type of multirobot system, typified by large numbers of homogeneous robots that interact implicitly with each other. Such systems are often contrasted with *heterogeneous* robots, in which team members may vary significantly in their capabilities. When robots vary in capabilities, challenges arise in determining which robots should perform which tasks – a challenge commonly referred to as *task allocation*. Finally *learning* in multirobot teams is of particular interest in designing teams that are adaptive over time and can learn new behaviors. Illustrating the advances in each of these areas often takes place in a set of representative application domains; these *applications* are the final major topic of discussion in this chapter.

40.2 Architectures for Multirobot Systems

The design of the overall control architecture for the multirobot team has a significant impact on the robustness and scalability of the system. Robot architectures for multirobot teams are composed of the same fundamental components as in single-robot systems, as described in Chap. 8. However, they also must address the inter-

action of robots and how the group behavior will be generated from the control architectures of the individual robots in the team. Several different philosophies for multirobot team architectures are possible; the most common are *centralized*, *hierarchical*, *decentralized*, and *hybrid*.

Centralized architectures that coordinate the entire team from a single point of control are theoretically possible [40.4], although often practically unrealistic due to their vulnerability to a single point of failure, and due to the difficulty of communicating the entire system state back to the central location at a frequency suitable for real-time control. Situations in which these approaches are relevant are cases in which the centralized controller has a clear vantage point from which to observe the robots, and can easily broadcast group messages for all robots to obey [40.5].

Hierarchical architectures are realistic for some applications. In this control approach, each robot oversees the actions of a relatively small group of other robots, each of which in turn oversees yet another group of robots, and so forth, down to the lowest robot, which simply executes its part of the task. This architecture scales much better than centralized approaches, and is reminiscent of military command and control. A point of weakness for the hierarchical control architecture is recovering from failures of robots high in the control tree.

Decentralized control architectures are the most common approach for multirobot teams, and typically require robots to take actions based only on knowledge local to their situation. This control approach can be highly robust to failure, since no robot is responsible for the control of any other robot. However, achieving global coherency in these systems can be difficult, because high-level goals have to be incorporated into the local control of each robot. If the goals change, it may be difficult to revise the behavior of individual robots.

Hybrid control architectures combine local control with higher-level control approaches to achieve both robustness and the ability to influence the entire team's actions through global goals, plans, or control. Many multirobot control approaches make use of hybrid architectures.

A plethora of multirobot control architectures have been developed over the years. We focus here on three approaches that illustrate the spectrum of control architectures. The first, the Nerd Herd, is representative of a pure *swarm* robotics approach using large numbers of homogeneous robots. The second, ALLIANCE, is representative of a behavior-based approach that enables coordination and control of possibly heterogeneous robots without explicit coordination. The third, distributed robot architecture (DIRA), is a hybrid approach that enables both robot autonomy and explicit coordination in possibly heterogeneous robot teams.

40.2.1 The Nerd Herd

One of the first studies of social behaviors in multi-robot teams was conducted by Mataric [40.6], with results being demonstrated on the Nerd Herd team of 20 identical robots (shown in Fig. 40.1). This work is an example of swarm robotic systems, as described further in Sect. 40.4. The decentralized control approach was based on the subsumption architecture (see Chap. 8), and assumed that all robots were homogeneous, but with relatively simple individual capabilities, such as detecting obstacles and *kin* (i. e., other robot team members). A set of basic social behaviors (see also Chap. 38) were defined and demonstrated, including obstacle avoidance, homing, aggregation, dispersion, following, and safe wandering. These basic behaviors were combined in various ways to yield more composite social behaviors, including flocking (composed of safe wandering, aggregation, and dispersion), surrounding (composed of safe wandering, following, and aggregation), herding (composed of safe wandering, surrounding, and flocking), and foraging (composed of safe wandering, dispersion, following, homing, and flocking). The behaviors were implemented as rules, such as the following rule for aggregate:

```
Aggregate:
  If agent is outside aggregation
    distance
      turn toward aggregation centroid
      and go.
  Else
    stop.
```

This work showed that collective behaviors could be generated through the combination of lower-level basic behaviors. Related work on this project studied



Fig. 40.1 The Nerd Herd robots

issues such as using bucket brigades to reduce interference [40.7], and learning [40.8].

40.2.2 The ALLIANCE Architecture

Another early work in multirobot team architectures is the ALLIANCE architecture (shown in Fig. 40.2), developed by Parker [40.9] for fault-tolerant task allocation in heterogeneous robot teams. This approach builds on the subsumption architecture by adding behavior sets and motivations for achieving action selection without explicit negotiations between robots. Behavior sets group low-level behaviors together for the execution of a particular task. The motivations consist of levels of impatience and acquiescence that can raise and lower a robot's interest in activating a behavior set corresponding to a task that must be accomplished.

In this approach, the initial motivation to perform a given behavior set is set to zero. Then, at each time step, the motivation level is recalculated based on

1. the previous motivation level
2. the rate of impatience
3. whether the sensory feedback indicates the behavior set is needed
4. whether the robot has another behavior set already activated
5. whether another robot has recently begun work on this task
6. whether the robot is willing to give up the task, based on how long it has been attempting the task

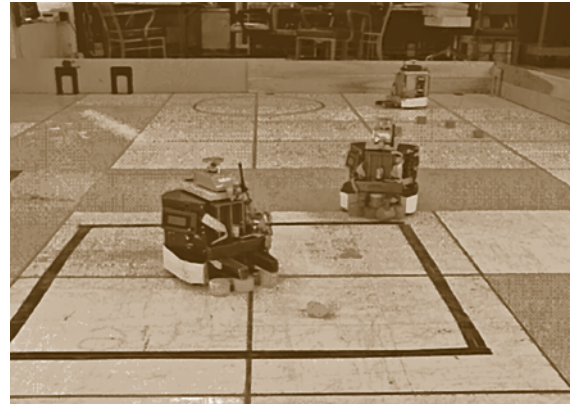


Fig. 40.3 Robots using the ALLIANCE architecture for a mock clean-up task

Effectively, the motivation continues to increase at some positive rate unless one of four situations occurs:

1. the sensory feedback indicates that the behavior set is no longer needed
2. another behavior set in the robot activates
3. some other robot has just taken over the task for the first time
4. the robot has decided to acquiesce the task

In any of these four situations, the motivation returns to zero. Otherwise, the motivation grows until it crosses a threshold value, at which time the behavior set is activated and the robot can be said to have selected an action. When an action is selected, cross-inhibition within that robot prevents other tasks from being activated within that same robot. When a behavior set is active in a robot, the robot broadcasts its current activity to other robots at a periodic rate.

The L-ALLIANCE extension [40.10] allows a robot to adapt the rate of change of the impatience and acquiescence values depending on the quality with which that robot is expected to accomplish a given task. The result is that robots that have demonstrated their ability to better accomplish certain tasks are more likely to choose those tasks in the future. Additionally, if problems occur during team performance, then robots may dynamically reallocate their tasks to compensate for the problems. This approach was demonstrated on a team of three heterogeneous robots performing a mock clean-up task, two robots performing a box-pushing task, and four robots performing a cooperative target observation problem. The approach has also been demonstrated in the simulation of a janitorial service task and a bound-

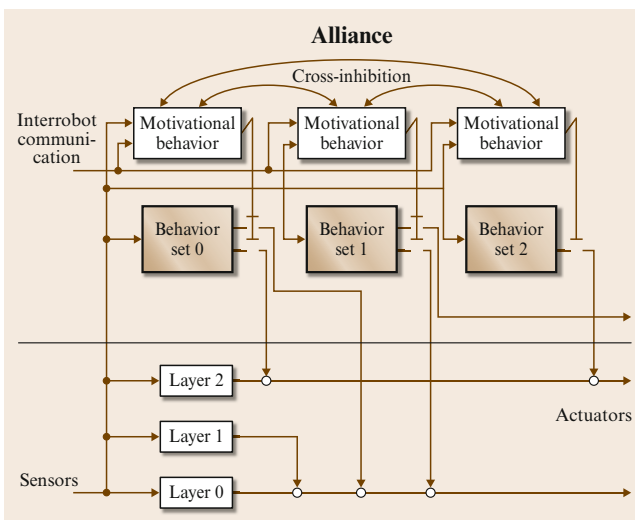


Fig. 40.2 The ALLIANCE architecture

ing overwatch task. Figure 40.3 shows robots using ALLIANCE to perform the mock clean-up task.

40.2.3 The Distributed Robot Architecture

Simmons et al. [40.11] have developed a hybrid architecture called the distributed robot architecture (DIRA). Similar to the Nerd Herd and ALLIANCE approaches, the DIRA approach allows autonomy in individual robots. However, unlike the previous approaches, DIRA also facilitates explicit coordination among robots. This approach is based on layered architectures that are popular for single-robot systems (see Chap. 8). In this approach (shown in Fig. 40.4), each robot's control architecture consists of a *planning* layer that decides how to achieve high-level goals; an *executive* layer that synchronizes agents, sequences tasks, and monitors task execution; and a *behavioral* layer that interfaces to the robot's sensors and effectors. Each of these layers interacts with those above and below it. Additionally, robots can interact with each other via direct connections at each of the layers.

This architecture has been demonstrated in a team of three robots – a crane, a roving eye, and a mobile manipulator – performing a construction assembly task (see Fig. 40.5). This task requires the robots to work together to connect a beam at a given location. In these demonstrations, a *foreman* agent decides which robot should move the beam at which times. Initially, the crane moves the beam to the vicinity of the emplacement based on encoder feedback. The foreman then sets up a behavioral loop between the roving eye and the crane robot to servo the beam closer to the point of emplacement. Once the beam is close enough, the foreman tasks the

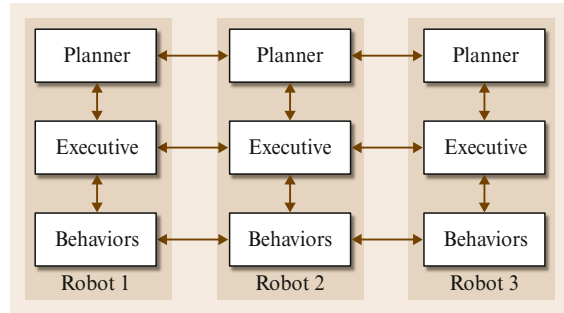


Fig. 40.4 The distributed robot architecture

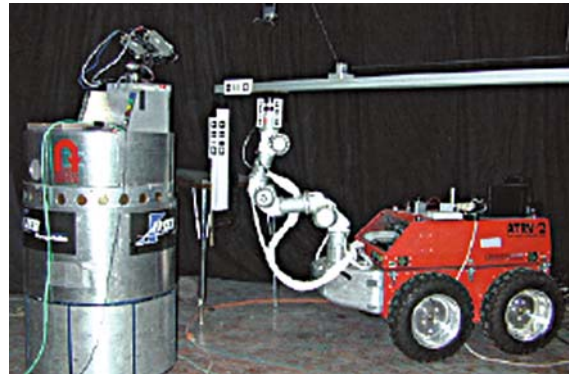


Fig. 40.5 Robots using the distributed robot architecture for assembly tasks

roving eye and the mobile manipulator to servo the arm to grasp the beam. After contact is made, the foreman tasks the roving eye and the mobile manipulator to coordinate to servo the beam to the emplacement point, thus completing the task.

40.3 Communication

A fundamental assumption in multirobot systems research is that globally coherent and efficient solutions can be achieved through the interaction of robots lacking complete global information. However, achieving these globally coherent solutions typically requires robots to obtain information about their teammates' states or actions. This information can be obtained in a number of ways; the three most common techniques are

1. the use of implicit communication *through the world* (called *stigmergy*), in which robots sense the effects of teammate's actions through their effects on the world (e.g., [40.6, 12–16])
2. passive action recognition, in which robots use sensors to directly observe the actions of their teammates (e.g., [40.17])
3. explicit (intentional) communication, in which robots directly and intentionally communicate relevant information through some active means, such as radio (e.g., [40.9, 18–20])

Each of these mechanisms for exchanging information between robots has its own advantages and disadvantages [40.21]. *Stigmergy* is appealing because of its simplicity and its lack of dependence upon explicit communications channels and protocols. However, it is

limited by the extent to which a robot's perception of the world reflects the salient states of the mission the robot team must accomplish. *Passive action recognition* is appealing because it does not depend upon a limited-bandwidth, fallible communication mechanism. As with implicit cooperation, however, it is limited by the degree to which a robot can successfully interpret its sensory information, as well as the difficulty of analyzing the actions of robot team members. Finally, the *explicit communication* approach is appealing because of its directness and the ease with which robots can become aware of the actions and/or goals of its teammates. The major uses of explicit communication in multirobot teams are to synchronize actions, exchange information, and to negotiate between robots. Explicit communication is a way of dealing with the *hidden-state* problem [40.22], in which limited sensors cannot distinguish between different states of the world that are important for task performance. However, explicit communication is limited in terms of fault tolerance and reliability, because it typically depends upon a noisy, limited-bandwidth communications channel that may not continually connect all members of the robot team. Thus, approaches that make use of explicit communications must also provide mechanisms to handle communication failures and lost messages.

Selecting the appropriate use of communication in a multirobot team is a design choice dependent upon the tasks to be achieved by the multirobot team. One needs to carefully consider the costs and benefits of alternative communications approaches to determine the method that can reliably achieve the required level of system performance. Researchers generally agree that communication can have a strong positive impact on the performance of the team. One of the earliest illustrations of this impact was given in the work of MacLennan [40.23], which investigates the evolution of communication in simulated worlds and concludes that

the communication of local robot information can result in significant performance improvements. Interestingly, for many representative applications, researchers have found a nonlinear relationship between the amount of information communicated and its impact on the performance of the team. Typically, even a small amount of information can have a significant impact on the team, as found in the study of Balch and Arkin [40.24]. However, more information does not necessarily continue to improve performance, as it can quickly overload the communications bandwidth without providing an application benefit. The challenge in multirobot systems is to discover the *optimal* pieces of information to exchange that yield these performance improvements without saturating the communications bandwidth. Currently, no general approaches to identifying this critical information are available; thus, the decision of what to communicate is an application-specific question to be answered by the system designer. Dudek's taxonomy of multirobot systems [40.25] includes axes related to communication, including communication range, communication topology, and communication bandwidth. These characteristics can be used to compare and contrast multirobot systems.

Several related issues of active research in communications for multirobot teams deal with dynamic network connectivity and topologies; for example, robot teams must either be able to maintain communications connectivity as they move, or employ recovery strategies that allow the robot team to recover when the communications connectivity is broken. These concerns may require robots to adapt their actions in response to the anticipated effects on the communications network, or in response to knowledge of the anticipated propagation behavior of information through the dynamic network. These and related issues are discussed in some detail in the context of *networked robotics*; see Chap. 41 for more information.

40.4 Swarm Robots

Historically, some of the earliest work in multirobot systems [40.12, 13, 26–33] dealt with large numbers of homogeneous robots, called *swarms*. Still undergoing active study today, the swarm approaches obtain inspiration from biological societies – particularly ants, bees, and birds – to develop similar behaviors in multirobot teams. Because biological societies are able to accomplish impressive group capabilities, such as the

ability of termites to build large complex mounds, or the ability of ants to collectively carry large prey, robotics researchers aim to reproduce these capabilities in robot societies.

Swarm robotics systems are often called *collective* robotics, indicating that individual robots are often unaware of the actions of other robots in the system, other than information on proximity. These approaches aim

to achieve a desired team-level global behavior from the interaction dynamics of individual robots following relatively simple local control laws. Swarm robotic systems typically involve very little explicit communication between robots, and instead rely on stigmergy (i. e., communication through the world) to achieve emergent cooperation. Individual robots are assumed to have minimal capabilities, with little ability to solve meaningful tasks on their own. However, when grouped with other similar robots, they are collectively able to achieve team-level tasks. Ideally, the entire team should be able to achieve much more than individual robots working alone (i. e., it is *superadditive*, meaning that the whole is bigger than the sum of the parts). These systems assume very large numbers of robots (at least dozens, and often hundreds or thousands) and explicitly address issues of scalability. Swarm robotic approaches achieve high levels of redundancy because robots are assumed to be identical, and thus interchangeable with each other.

Many types of swarm behaviors have been studied, such as foraging, flocking, chaining, search, herding, aggregation, and containment. The majority of these swarm behaviors deal with spatially distributed multi-robot motions, requiring robots to coordinate motions either

1. relative to other robots
2. relative to the environment
3. relative to external agents

4. relative to robots and the environment
5. relative to all (i. e., other robots, external agents, and the environment)

Table 40.1 categorizes swarm robot behaviors according to these groupings, citing representative examples of relevant research.

Much of the current research in swarm robotics is aimed at developing specific solutions to one or more of the swarm behaviors listed in Table 40.1. Some of these swarm behaviors have received particular attention, notably formations, flocking, search, coverage, and foraging. Section 40.8 discusses these behaviors in more detail. In general, most current work in the development of swarm behaviors is aimed not just at demonstrating group motions that are similar to biological systems, but also at understanding the formal control theoretic principles that can predictably converge to the desired group behaviors, and remain in stable states.

Demonstration of physical robot swarms is both a hardware and a software challenge. As discussed in Sect. 40.2, the first demonstrations were by *Mataric* [40.6], involving about 20 physical robots performing aggregation, dispersion, and flocking. This work defined composable *basis behaviors* as primitives for structuring more complex systems (see Chap. 38 for more information). More recently, *McLurkin* [40.50] developed an extensive catalog of swarm behavior software, and demonstrated these behaviors on about 100 physical robots (called the

Table 40.1 Categories of swarm behaviors

Relative motion requirements	Swarm behaviors
Relative to other robots	Formations [40.34, 35], flocking [40.29], natural herding (as in herds of cattle), schooling, sorting [40.14], clumping [40.14], condensation, aggregation [40.36], dispersion [40.37]
Relative to the environment	Search [40.38], foraging [40.39], grazing, harvesting, deployment [40.40], coverage [40.41], localization [40.42], mapping [40.43], exploration [40.44]
Relative to external agents	Pursuit [40.45], predator–prey [40.46], target tracking [40.47], forced herding/shepherding (as in shepherding sheep)
Relative to other robots and the environment	Containment, orbiting, surrounding, perimeter search [40.48]
Relative to other robots, external agents, and the environment	Evasion, tactical overwatch, soccer [40.49]

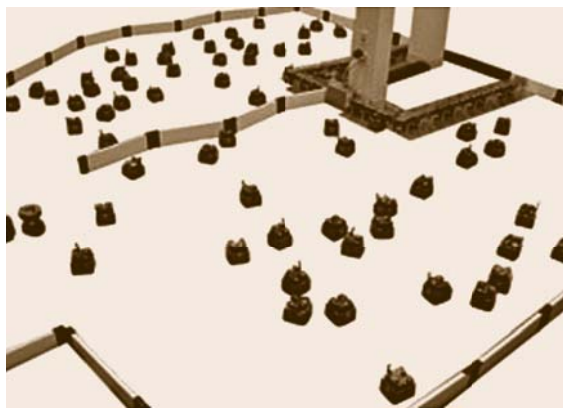


Fig. 40.6 The SwarmBot robots

SwarmBot robots), developed by iRobot, as shown in Fig. 40.6. He created several group behaviors, such as `avoidManyRobots`, `disperseFromSource`, `disperseFromLeaves`, `disperseUniformly`, `computeAverageBearing`, `avoidManyRobots`, `followTheLeader`, `orbitGroup`, `navigateGradient`, `clusterOnSource`, and `clusterIntoGroups`. A swarm of 108 robots used the developed dispersion algorithms in an empty schoolhouse of area of about 300 m², and were able to locate an object of interest and lead a human to its location [40.37].

The European Union has sponsored several swarm robot projects, leading toward decreasingly smaller sized individual robots. The I-SWARM project, for instance, is aimed at developing millimeter-sized robots

with full onboard sensing, computation, and power for performing biologically inspired swarming behaviors, as well as collective perception tasks. This project is both a hardware and a software challenge, in that developing microscale robots that are fully autonomous and can perform meaningful cooperative behaviors will require significant advances in the current state of the art.

Another notable effort in swarm robotics research is the US multi-university SWARMS initiative led by the University of Pennsylvania. Research in this project is aimed at developing a new system-theoretic framework for swarming, developing models of swarms and swarming behavior, analyzing swarm formation, stability, and robustness, synthesizing emergent behaviors for active perception and coverage, and developing algorithms for distributed localization.

Besides the hardware challenges of dealing with large numbers of small robots, there are many important software challenges that remain to be solved. From a practical perspective, the usual approach to creating homogeneous multirobot swarms is to hypothesize a possible local control law (or laws), and then study the resulting group behavior, iterating until the desired global behavior is obtained. However, the longer-term objective is to be able to both predict group performance based on known local control laws, and to generate local control laws based upon a desired global group behavior. Active research by many investigators is ongoing to develop solutions to these key research challenges.

40.5 Heterogeneity

Robot heterogeneity can be defined in terms of variety in robot behavior, morphology, performance quality, size, and cognition. In most large-scale multirobot systems work, the benefits of parallelism, redundancy, and solutions distributed in space and time are obtained through the use of homogeneous robots, which are completely interchangeable (i.e., the swarm approach, as described in Sect. 40.4). However, certain complex applications of large-scale robot teams may require the simultaneous use of multiple types of sensors and robots, all of which cannot be designed into a single type of robot. Some robots may need to be scaled to smaller sizes, which will limit their payloads, or certain required sensors may be too expensive to duplicate across all robots on the team. Other robots may need to be large to carry application-specific

payload or sensors, or to navigate long distances in a limited time. These applications, therefore, require the collaboration of large numbers of heterogeneous robots.

The motivation for developing heterogeneity in multirobot teams is thus twofold: heterogeneity may be a design feature beneficial to particular applications, or heterogeneity may be a necessity. As a design feature, heterogeneity can offer economic benefits, since it can be easier to distribute varying capabilities across multiple team members rather than to build many copies of monolithic robots. Heterogeneity can also offer engineering benefits, as it may simply be too difficult to design individual robots that incorporate all of the sensing, computational, and effector requirements of a given application. Heterogeneity in behavior may also arise in

an emergent manner in physically homogeneous teams, as a result of behavior specialization.

A second compelling reason to study heterogeneity is that it may be a necessity, in that it is nearly impossible in practice to build a truly homogeneous robot team. The realities of individual robot design, construction, and experience will inevitably cause a multirobot system to drift to heterogeneity over time. This is recognized by experienced roboticists, who have seen that several copies of the same model of robot can vary widely in capabilities due to differences in sensor tuning, calibration, etc. Over time, even minor initial differences among robots will grow due to individual robot drift and wear and tear. The implication is that, to employ robot teams effectively, we must understand diversity, predict how it will impact performance, and enable robots to adapt to the diverse capabilities of their peers. In fact, it is often advantageous to build diversity explicitly into the design of a robot team.

There are a variety of research challenges in heterogeneous multirobot systems. A particular challenge to achieving efficient autonomous control is when overlap in team member capabilities occurs, thus affecting task allocation or role assignments [40.51]. Techniques as described in Sect. 40.6 can typically deal with heterogeneous robots for the purposes of task allocation. Another important topic in heterogeneity is how to recognize and quantify heterogeneity in multirobot teams. Some types of heterogeneity can be evaluated quantitatively, using metrics such as the social entropy metric developed by Balch [40.52]. Most research in heterogeneous multirobot systems assumes that robots have a common language and a common understanding of symbols in their language; developing a common understanding of communicated symbols among robots with different physical capabilities is a fundamental challenge, addressed by Jung in [40.53].

As discussed in Sect. 40.2, one of the earliest research demonstrations of heterogeneity in physical robot teams was in the development of the ALLIANCE architecture by Parker [40.9]. This work demonstrated the ability of robots to compensate for heterogeneity in team members during task allocation and execution. Murphy has studied heterogeneity in the context of marsupial robot deployment, where a *mothership* robot assists smaller robots in applications such as search and rescue [40.54]. Grabowski et al. [40.43] developed modular millibots for surveillance and reconnaissance that could be composed of interchangeable sensor and effector components, thus creating a variety of different heterogeneous teams. Simmons et al. [40.11]



Fig. 40.7 Heterogeneous team of an air and two ground vehicles that can perform cooperative reconnaissance and surveillance

demonstrated the use of heterogeneous robots for autonomous assembly and construction tasks relevant to space applications. Sukatme et al. [40.55] demonstrated a helicopter robot cooperating with two ground robots in tasks involving marsupial-inspired payload deployment and recovery, cooperative localization, and reconnaissance and surveillance tasks, as shown in Fig. 40.7. Parker et al. [40.56] demonstrated assistive navigation for sensor network deployment using a more intelligent leader robot for guiding navigationally challenged simple sensor robots to goal locations, as part of a larger demonstration by Howard et al. [40.57] of 100 robots performing exploration, mapping, deployment, and detection. Chaimowicz et al. [40.58] demonstrated a team of aerial and ground robots cooperating for surveillance applications in urban environments. Parker and Tang [40.59] developed ASyMTRe (Automated Synthesis of Multirobot Task solutions through software Reconfiguration), which enables heterogeneous robots to share sensory resources to enable the team to accomplish tasks that would be impossible without tightly coupled sensor sharing.

Many open research issues remain to be solved in heterogeneous multirobot teams; for example, the issue of optimal team design is a very challenging problem. Clearly, the required behavioral performance in a given application dictates certain constraints on the physical design of the robot team members. However, it is also clear that multiple choices may be made in designing a solution to a given application, based upon cost, robot availability, ease of software design, flexibility in robot use, and so forth. Designing an optimal robot team for a given application requires significant analysis and consideration of the tradeoffs in alternative strategies.

40.6 Task Allocation

In many multirobot applications, the mission of the team is defined as a set of tasks that must be completed. Each task can usually be worked on by a variety of different robots; conversely, each robot can usually work on a variety of different tasks. In many applications, a task is decomposed into independent subtasks [40.9], hierarchical task trees [40.60], or roles [40.11, 58, 61, 62] either by a general autonomous planner or by the human designer. Independent subtasks or roles can be achieved concurrently, while subtasks in task trees are achieved according to their interdependence. Once the set of tasks or subtasks have been identified, the challenge is to determine the preferred mapping of robots to tasks (or subtasks). This is the *task allocation* problem.

The details of the task allocation problem can vary in many dimensions, such as the number of robots required per task, the number of tasks a robot can work on at a time, the coordination dependencies among tasks, and the time frame for which task assignments are determined. *Gerkey and Mataric* [40.63] defined a taxonomy for task allocation that provides a way of distinguishing task allocation problems along these dimensions, which is referred to as the multirobot task allocation (**MRTA**) taxonomy.

40.6.1 Taxonomy for Task Allocation

Generally, tasks are considered to be of two principal types: *single-robot tasks* (**SR**, according to the **MRTA** taxonomy) are those that require only one robot at a time, while *multirobot tasks* (**MR**) are those that require more than one robot working on the same task at the same time. Commonly, single-robot tasks that have minimal task interdependencies are referred to as *loosely coupled tasks*, representing a *weakly cooperative* solution. On the other hand, multirobot tasks are often considered to be sets of subtasks that have strong interdependencies. These tasks are therefore often referred to as *tightly coupled tasks* that require a *strongly cooperative* solution. The subtasks of a loosely coupled multirobot task require a high level of synchronization or coordination between subtasks, meaning that each task must be aware of the current state of the coordinated subtasks within a small time delay. As this time delay becomes progressively larger, coordinated subtasks become more loosely coupled, representing weakly cooperative solutions.

Robots can also be categorized as either *single-task robots* (**ST**), which work on only one task at a time or *multitask robots* (**MT**), which are able to make progress

on more than one task at a time. Most commonly, task allocation problems assume robots are single-task robots, since more capable robots that perform multiple tasks in parallel are still beyond the current state of the art.

Tasks can either be assigned to optimize the instantaneous allocation of tasks (**IA**), or to optimize the assignments into the future (**TA**, for time-extended assignment). In the case of instantaneous assignment, no consideration is made for the effect of the current assignment on future assignments. Time-extended assignments attempt to assign tasks so that the performance of the team is optimized for the entire set of tasks that may be required, not just the current set of tasks that need to be achieved at the current time step.

Using the MTRA taxonomy, triples of these abbreviations are used to categorize various task allocation approaches, such as **SR-ST-IA**, which refers to an assignment problem in which single-robot tasks are assigned once to single-task robots. Different variations of the task allocation problem have different computational complexities. The easiest variant is the **ST-SR-IA** problem, which can be solved in polynomial time since it is an instance of the optimal assignment problem [40.64]. Other variants are much more difficult, and do not have known polynomial time solutions. For example, the **ST-MR-IA** variant can be shown to be an instance of the set partitioning problem [40.65], which is strongly *NP*-hard. The **ST-MR-TA**, **MT-SR-IA**, and **MT-SR-TA** variants have also all been shown to be *NP*-hard problems. Because these problems are computationally complex, most approaches to task allocation in multirobot teams generate approximate solutions.

40.6.2 Representative Approaches

Approaches to task allocation in multirobot teams can be roughly divided into behavior-based approaches and market-based (sometimes called negotiation-style or auction-based) approaches. The following subsections describe some representative architectures for each of these general approaches. Refer to [40.63] for a comparative analysis of some of these approaches, in terms of computation and communications requirements and solution quality.

Behavior-Based Task Allocation

Behavior-based approaches typically enable robots to determine task assignments without explicitly discussing individual tasks. In these approaches, robots use

knowledge of the current state of the robot team mission, robot team member capabilities, and robot actions to decide, in a distributed fashion, which robot should perform which task.

One of the earliest architectures for multirobot task allocation that was demonstrated on physical robots was the behavior-based ALLIANCE architecture [40.9] and the related L-ALLIANCE architecture [40.10]. ALLIANCE addresses the **ST-SR-IA** and **ST-SR-TA** variants of the task allocation problem without explicit communication among robots about tasks. As described in Sect. 40.2.2, ALLIANCE achieves adaptive action selection through the use of motivational behaviors, which are levels of impatience and acquiescence within each robot that determine its own and its teammates' relative fitness for performing certain tasks. These motivations are calculated based upon the mission requirements, the activities and capabilities of teammates, and the robots' internal states. These motivations effectively calculate utility measures for each robot–task pair.

Another behavior-based approach to multirobot task allocation is broadcast of local eligibility (**BLE**) [40.66], which addresses the **ST-SR-IA** variant of task allocation. **BLE** uses a subsumption style behavior control architecture [40.67] that allows robots to efficiently execute tasks by continuously broadcasting locally computed eligibilities and only selecting the robot with the best eligibility to perform the task. In this case, task allocation is achieved through behavior inhibition. **BLE** uses an assignment algorithm that is very similar to *Botelho* and *Alami*'s **M+** architecture [40.68].

Market-Based Task Allocation

Market-based (or negotiation-based) approaches typically involve explicit communications between robots about the required tasks, in which robots bid for tasks based on their capabilities and availability. The negotiation process is based on market theory, in which the team seeks to optimize an objective function based upon individual robot utilities for performing particular tasks. The approaches typically greedily assign subtasks to the robot that can perform the task with the highest utility.

Smith's contract net protocol (CNP) [40.69] was the first to address the problem of how agents can negotiate to collectively solve a set of tasks. The use of a market-based approach specifically for multirobot task allocation was first developed by *Botelho* and *Alami* with their **M+** architecture [40.68]. In the **M+** approach, robots plan their own individual plans for the task they have been assigned. They then negotiate with other teammates to incrementally adapt their actions to suit the

team as a whole, through the use of social rules that facilitate the merging of plans.

Since these early developments, many alternative approaches to market-based task allocation have been developed. A thorough survey on the current state of the art in market-based techniques for multirobot task allocation is given in [40.70], comparing alternative approaches in terms of solution quality, scalability, dynamic events and environments, and heterogeneous teams.

Most of the current approaches in market-based task allocation address the **ST-SR** problem variant, with some approaches (e.g., [40.11, 71–73]) dealing with instantaneous assignment (**IA**), and others (e.g., [40.44, 74–76]) addressing time-extended assignments (**TA**). More recent methods are beginning to address the allocation of multirobot tasks (i.e., the **MR-ST** problem variant), including [40.59, 77–81]. An example approach to the **MR-MT** problem variant is found in [40.82].

Some representative market-based techniques include **MURDOCH** [40.71], **TraderBots** [40.60, 76], and **Hoplites** [40.78]. The **MURDOCH** approach [40.71] employs a resource-centric, publish–subscribe communication model to carry out auctions, which has the advantage of anonymous communication. In this approach a task is represented by the required resources, such as the environmental sensors. The methods for how to use such a sensor to generate satisfactory results is preprogrammed into the robot.

The **TraderBots** approach [40.60, 76] applies market economy techniques for generating efficient and robust multirobot coordination in dynamic environments. In a market economy, robots act based on selfish interests. A robot receives revenue and incurs cost when trying to accomplish a task. The goal is for robots to trade tasks through auctions/negotiations such that the team profit (revenue minus cost) is optimized.

The **Hoplites** approach [40.78] focuses on the selection of an appropriate joint plan for the team to execute by incorporating joint revenue and cost into the bid. This approach couples planning with passive and active coordination strategies, enabling robots to change coordination strategies as the needs of the task change. Strategies are predefined for a robot to accomplish a selected plan.

Some alternative approaches formulate the objects to be assigned as *roles*, which typically package a set of tasks and/or behaviors that a robot should undertake when acting in a particular role. Roles can then be dynamically assigned to robots in a similar manner as in the auction-based approaches (e.g., [40.11, 61]).

40.7 Learning

Multirobot learning is the problem of learning new cooperative behaviors, or learning in the presence of other robots. The other robots in the environment, however, have their own goals and may be learning in parallel [40.83]. The challenge is that having other robots in the environment violates the Markov property that is a fundamental assumption of single-robot learning approaches [40.83]. The multirobot learning problem is particularly challenging because it combines the difficulties of single-robot learning with multiagent learning. Particular difficulties that must be considered in multirobot learning include continuous state and action spaces, exponential state spaces, distributed credit assignment, limited training time and insufficient training data, uncertainty in sensing and shared information, nondeterministic actions, difficulty in defining appropriate abstractions for learned information, and difficulty of merging information learned from different robot experiences.

The types of applications that have been studied for multirobot learning include multitarget observation [40.84, 85], air fleet control [40.86], predator-prey [40.46, 87, 88], box pushing [40.89], foraging [40.22], and multirobot soccer [40.49, 90]. Particularly challenging domains for multirobot learning are those tasks that are *inherently* cooperative. Inherently cooperative tasks are those that cannot be decomposed into independent subtasks to be solved by individual robots. Instead, the utility of the action of one robot is dependent upon the current actions of the other team members. This type of task is a particular challenge in multirobot learning, due to the difficulty of assigning credit for the individual actions of the robot team members.

The credit assignment problem is a particular challenge, since it is difficult for a robot to determine whether the fitness (either good or bad) is due to its own actions, or due to the actions of another robot. As discussed by *Pugh* and *Martinoli* in [40.91], this problem can be especially difficult in situations where robots do not explicitly share their intentions. Two different variations of the credit assignment problem are common in multirobot learning. The first is when robots are learning individual behaviors in the presence of other robots that can affect their performance. The second is when robots are attempting to learn a task with

a shared fitness function. It can be difficult to determine how to decompose the fitness function to appropriately reward or penalize the contributions of individual robots.

While learning has been explored extensively in the area of single-robot systems (see, for example, the discussion of learning in behavior-based systems in Chap. 38, and a discussion of fundamental learning techniques in Chap. 9) and in multiagent systems [40.92], much less work has been done in the area of multirobot learning, although the topic is gaining increased interest. Much of the work to date has focused on reinforcement learning approaches. Some examples of this multirobot learning research include the work by *Asada* et al. [40.93], who propose a method for learning new behaviors by coordinating previously learned behaviors using Q-learning, and apply it to soccer-playing robots. *Matarić* [40.8] introduces a method for combining basic behaviors into higher-level behaviors through the use of unsupervised reinforcement learning, heterogeneous reward functions, and progress estimators. This mechanism was applied to a team of robots learning to perform a foraging task. *Kubo* and *Kakazu* [40.94] proposed another reinforcement learning mechanism that uses a progress value for determining reinforcement, and applied it to simulated ant colonies competing for food. *Fernandez* et al. [40.84] apply a reinforcement learning algorithm that combines supervised function approximation with generalization methods based on state-space discretization, and apply it to robots learning the multiobject tracking problem. *Bowling* and *Veloso* [40.83] developed a general-purpose, scalable learning algorithm called GraWoLF (Gradient-based Win or Learn Fast), which combines gradient-based policy learning techniques with a variable learning rate, and demonstrated the results in the adversarial multirobot soccer application.

Other multirobot learning approaches not based on reinforcement include *Parker's* L-ALLIANCE architecture [40.10], which uses parameter tuning, based on statistical experience data, to learn the fitness of different heterogeneous robots in performing a set of tasks. *Pugh* and *Martinoli* [40.91] apply particle swarm optimization techniques to distributed unsupervised robot learning in groups, for the task of learning obstacle avoidance.

40.8 Applications

Many real-world applications can potentially benefit from the use of multiple mobile robot systems. Example applications include container management in ports [40.95], extraplanetary exploration [40.96], search and rescue [40.54], mineral mining, transportation, industrial and household maintenance, construction [40.11], hazardous waste cleanup [40.9], security [40.97, 98], agriculture, and warehouse management [40.99]. Multiple robot systems are also used in the domain of localization, mapping, and exploration; Chap. 37 mentions some of the work in multirobot systems applied to these problems. Part F of this Handbook outlines many application areas that are relevant not only to single-robot systems, but also to multiple mobile robot systems. To date, relatively few real-world implementations of these multirobot systems have occurred, primarily due to the complexities of multiple robot systems and the relative newness of the supporting technologies. Nevertheless, many proof-of-principle demonstrations of physical multirobot systems have been achieved, and the expectation is that these systems will find their way into practical implementations as the technology continues to mature.

Research in multiple mobile robot systems is often explored in the context of common application test domains. While not yet elevated to the level of benchmark tasks, these common domains do provide opportunities for researchers to compare and contrast alternative strategies to multirobot control. Additionally, even though these common test domains are usually just laboratory experiments, they do have relevance to real-world applications. This section outlines these common application domains; see also [40.2] and [40.100] for a discussion of these domains and a more detailed listing of related research.

40.8.1 Foraging and Coverage

Foraging is a popular testing application for multirobot systems, particularly for those approaches that address swarm robotics, involving very large numbers of mobile robots. In the foraging domain, objects such as pucks or simulated food pellets are distributed across the planar terrain, and robots are tasked with collecting the objects and delivering them to one or more gathering locations, such as a home base. Foraging lends itself to the study of weakly cooperative robot systems, in that the actions of individual robots do not

have to be tightly synchronized with each other. This task has traditionally been of interest in multirobot systems because of its close analogy to the biological systems that motivate swarm robotics research. However, it also has relevance to several real-world applications, such as toxic waste cleanup, search and rescue, and demining. Additionally, since foraging usually requires robots to completely explore their terrain in order to discover the objects of interest, the *coverage* domain has similar issues to the foraging application. In coverage, robots are required to visit all areas of their environment, perhaps searching for objects (such as landmines) or executing some action in all parts of the environment (e.g., for floor cleaning). The coverage application also has real-world relevance to tasks such as demining, lawn care, environmental mapping, and agriculture.

In foraging and coverage applications, a fundamental question is how to enable the robots to explore their environments quickly without duplicating actions or interfering with each other. Alternative strategies can include basic stigmergy [40.14], forming chains [40.28], and making use of heterogeneous robots [40.39]. Other research demonstrated in the foraging and/or coverage domain includes [40.22, 41, 101–106].

40.8.2 Flocking and Formations

Coordinating the motions of robots relative to each other has been a topic of interest in multiple mobile robot systems since the inception of the field. In particular, much attention has been paid to the flocking and formation control problems. The flocking problem could be viewed as a subcase of the formation control problem, requiring robots to move together along some path in the aggregate, but with only minimal requirements for paths taken by specific robots. Formations are stricter, requiring robots to maintain certain relative positions as they move through the environment. In these problems, robots are assumed to have only minimal sensing, computation, effector, and communications capabilities. A key question in both flocking and formation control research is determining the design of local control laws for each robot that generate the desired emergent collective behavior. Other issues include how robots cooperatively localize themselves to achieve formation control (e.g., [40.42, 107]), and how paths can be planned for permutation-invariant multirobot formations (e.g., [40.108]).

Early solutions to the flocking problem in artificial agents were generated by *Reynolds* [40.109] using a rule-based approach. Similar behavior- or rule-based approaches have been used physical robot demonstrations and studies, such as in [40.29, 110]. These earlier solutions were based on human-generated local control rules that were demonstrated to work in practice. More recent work is based on control theoretic principles, with a focus on proving stability and convergence properties in multirobot team behaviors. Examples of this work include [40.36, 111–119].

40.8.3 Box Pushing and Cooperative Manipulation

Box pushing and cooperative manipulation are popular domains for demonstrating multirobot cooperation, because they offer a clear domain where close coordination and cooperation is required. Box pushing requires robot teams to move boxes from their starting positions to defined goal configurations, sometimes along specified paths. Typically, box pushing operates in the plane, and the assumption is made that the boxes are too heavy or too long to enable single robots to push alone. Sometimes there are several boxes to be moved, with ordering dependencies constraining the sequence of motions. Cooperative manipulation is similar, except it requires robots to lift and carry objects to a destination. This test bed domain lends itself to the study of strongly cooperative multirobot strategies, since robots often have to synchronize their actions to successfully execute these tasks. The domain of box pushing and cooperative manipulation is also popular because it has relevance to several real-world applications [40.100], including warehouse stocking, truck loading and unloading, transporting large objects in industrial environments, and assembly of large-scale structures.

Researchers usually emphasize different aspects of their cooperative control approach in the box pushing and cooperative manipulation domain. For example, *Kube and Zhang* [40.13] demonstrate how swarm-type cooperative control techniques could achieve box pushing, *Parker* [40.10, 120] illustrates aspects of adaptive task allocation and learning, *Donald et al.* [40.121] illustrates concepts of information invariance and the interchangeability of sensing, communication, and control, and *Simmons et al.* [40.11] demonstrate the feasibility of cooperative control for building planetary habitats. A significant body of additional research has been illustrated in this domain; representative examples include [40.3, 6, 31, 71, 96, 122–130].

40.8.4 Multitarget Observation

The domain of multitarget observation requires multiple robots to monitor and/or observe multiple targets moving through the environment. The objective is to maximize the amount of time, or the likelihood, that the targets remain in view by some team member throughout task execution. The task can be especially challenging if there are more targets than robots. This application domain can be useful for studying strongly cooperative task solutions, since robots have to coordinate their motions or the switching of targets to follow in order to maximize their objective. In the context of multiple mobile robot applications, the planar version of this test bed was first introduced in [40.131] as cooperative multirobot observation of multiple moving targets (CMOMMT). Similar problems have been studied by several researchers, and extended to more complex problems such as environments with complex topography or three-dimensional versions for multiple aerial vehicle applications. This domain is also related to problems in other areas, such as art gallery algorithms, pursuit evasion, and sensor coverage. This domain has practical application in many security, surveillance, and reconnaissance problems. Research applied to the multitarget observation problem in multirobot systems includes [40.47, 66, 132–136].

40.8.5 Traffic Control and Multirobot Path Planning

When multiple robots are operating in a shared environment, they must coordinate their actions to prevent interference. These problems typically arise when the space in which robots operate contains bottlenecks, such as networks of roadways, or when the robots take up a relatively large portion of the navigable space. In these problems, the open space can be viewed as a resource that robots must share as efficiently as possible, avoiding collisions and deadlocks. In this domain, robots usually have their own individual goals, and must work with other robots to ensure that they receive use of the shared space to the extent needed to achieve their goals. In some variants, the entire paths of multiple robots need to be coordinated with each other; in other variants, robots must simply avoid interfering with each other.

A variety of techniques have been introduced to address this problem, including traffic rules, subdividing the environment into single-ownership sections, and geometric path planning. Many of the earliest research approaches to this problem were based on heuristic approaches, such as predefining motion control (or traffic)

rules that were shown to prevent deadlock [40.137–140], or using techniques similar to mutual exclusion in distributed computing [40.141, 142]. These approaches have the benefit of minimizing the planning cost for obtaining a solution. Other, more formal, techniques view the application as a geometric multirobot path planning problem that can be solved precisely in configuration space–time. Chapter 5 includes a discussion of motion planning for multiple robots relevant to this domain. While geometric motion planning approaches provide the most general solutions, they can often be too computationally intensive for practical application, impractical due to the dynamic nature of the environment, or simply unnecessary for the problem at hand. In these cases, heuristic approaches may be sufficient.

40.8.6 Soccer

Since the inception of the RoboCup multirobot soccer domain as a proposed challenge problem for studying coordination and control in multirobot systems [40.143], research in this domain has grown tremendously. This domain incorporates many challenging aspects of multirobot control, including collaboration, robot control architectures, strategy acquisition, real-time reasoning and action, sensor fusion, dealing with adversarial environments, cognitive modeling, and learning. Annual competitions show the ever-improving team capabilities of the robots in a variety of settings, as shown



Fig. 40.8 Legged robot teams competing in robot soccer

in Fig. 40.8. A key aspect of this domain that is not present in the other multirobot test domains is that robots must operate in *adversarial* environments. This domain is also popular because of its educational benefits, as it brings together students and researchers from across the world in competitions to win the RoboCup challenges. The RoboCup competitions have added an additional search-and-rescue category to the competition [40.144], which has also become a significant area of research (see Chap. 50 for more details on this field). Annual proceedings of the RoboCup competitions document much of the research that is incorporated into the multirobot soccer teams. Some representative research works include [40.145–149].

40.9 Conclusions and Further Reading

This chapter has surveyed the current state of the art in multirobot systems, examining architectures, communications issues, swarm robot systems, heterogeneous teams, task allocation, learning, and applications. Clearly, significant advances have been made in the field in the last decade. The field is still an active area of research, however, since many open research issues still remain to be solved. Key open research questions remain in the broad areas of system integration, robustness, learning, scalability, generalization, and dealing with heterogeneity.

For example, in the area of system *integration*, an open question is how to effectively allow robot teams to combine a spectrum of approaches toward achieving complete systems that can perform more than a limited set of tasks. In the area of *robustness*, multirobot teams still need improvements in the ability to degrade

gracefully, to reason for fault tolerance, and to achieve complexity without escalating failure rates. The area of *learning* in multirobot teams is still in its infancy, with open questions including how to achieve continual learning in multirobot teams, how to facilitate the use of complex representations, and how to enable humans to influence and/or understand the results of the team learning. *Scalability* is still a challenging problem, in terms of more complex environments as well as ever-larger numbers of robots. Open issues in *generalization* include enabling the robot team to reason about context and increasing the versatility of systems so that they can operate in a variety of different applications. In dealing with *heterogeneity*, open questions include determining theoretical approaches to predicting system performance when all robots are not equal, and determining how to design a robot team optimally for a given application.

These problems, and others, promise to keep the field of multiple mobile robot systems active for many years to come.

For further reading on the topic of multiple mobile robot systems, the reader is referred to survey articles in the field, including [40.2, 100, 150, 151]. Additionally, several special journal issues on this topic have

appeared, including [40.1, 152–154]. Some taxonomies of multirobot systems are given in [40.25, 100, 155]. A variety of symposia and workshops have been held on a regular basis on the topic of multirobot systems; recent proceedings of these workshops and symposia include [40.156–163]. An additional edited text on this topic is [40.164].

References

- 40.1 T. Arai, E. Pagello, L.E. Parker: Editorial: Advances in multi-robot systems, *IEEE Trans. Robot. Autom.* **18**(5), 655–661 (2002)
- 40.2 Y. Cao, A. Fukunaga, A. Kahng: Cooperative mobile robotics: Antecedents and directions, *Auton. Robot.* **4**, 1–23 (1997)
- 40.3 R.G. Brown, J.S. Jennings: A pusher/steerer model for strongly cooperative mobile robot manipulation, *Proc. IEEE Int. Conf. Intell. Robot. Syst. (IROS '95)* (IEEE, Pittsburgh 1995) pp. 562–568
- 40.4 D. Milutinović, P. Lima: Modeling and optimal centralized control of a large-size robotic population, *IEEE Trans. Robot.* **22**(6), 1280–1285 (2006)
- 40.5 B. Khoshnevis, G.A. Bekey: Centralized sensing and control of multiple mobile robots, *Comput. Ind. Eng.* **35**(3–4), 503–506 (1998)
- 40.6 M.J. Mataríć: Issues and approaches in the design of collective autonomous agents, *Robot. Auton. Syst.* **16**, 321–331 (1995)
- 40.7 E. Ostergaard, G.S. Sukhatme, M.J. Mataríć: Emergent bucket brigading, *Fifth International Conference on Autonomous Agents* (Montreal 2001)
- 40.8 M. Mataríć: Reinforcement learning in the multi-robot domain, *Auton. Robot.* **4**, 73–83 (1997)
- 40.9 L.E. Parker: ALLIANCE: An Architecture for Fault-Tolerant Multi-Robot Cooperation, *IEEE Trans. Robot. Autom.* **14**(2), 220–240 (1998)
- 40.10 L.E. Parker: Lifelong adaptation in heterogeneous teams: Response to continual variation in individual robot performance, *Auton. Robot.* **8**(3), 239–267 (2000)
- 40.11 R. Simmons, S. Singh, D. Hershberger, J. Ramos, T. Smith: First results in the coordination of heterogeneous robots for large-scale assembly, *Proc. ISER 7th Int. Symp. Exp. Robot.* (Springer, New York 2000)
- 40.12 J. Deneubourg, S. Goss, G. Sandini, F. Ferrari, P. Dario: Self-organizing collection and transport of objects in unpredictable environments. (Kyoto) pp. 1093–1098
- 40.13 C.R. Kube, H. Zhang: Collective robotics: From social insects to robots, *Adapt. Behav.* **2**(2), 189–219 (1993)
- 40.14 R. Beckers, O. Holland, J. Deneubourg: From local actions to global tasks: Stigmergy and collective robotics, *Proc. 14th Int. Workshop Synth. Simul. Living Syst.*, ed. by R. Brooks, P. Maes (MIT Press, Cambridge 1994) pp. 181–189
- 40.15 S. Onn, M. Tennenholtz: Determination of social laws for multi-agent mobilization, *Artif. Intell.* **95**, 155–167 (1997)
- 40.16 B.B. Werger: Cooperation without deliberation: A minimal behavior-based approach to multi-robot teams, *Artif. Intell.* **110**(2), 293–320 (1999)
- 40.17 M.J. Huber, E. Durfee: Deciding when to commit to action during observation-based coordination, *Proc. 1st Int. Conf. Multi-Agent Syst.* (1995) pp. 163–170
- 40.18 H. Asama, K. Ozaki, A. Matsumoto, Y. Ishida, I. Endo: Development of task assignment system using communication for multiple autonomous robots, *J. Robot. Mechatron.* **4**(2), 122–127 (1992)
- 40.19 N. Jennings: Controlling cooperative problem solving in industrial multi-agent systems using joint intentions, *Artif. Intell.* **75**(2), 195–240 (1995)
- 40.20 M. Tambe: Towards flexible teamwork, *J. Artif. Intell. Res.* **7**, 83–124 (1997)
- 40.21 L.E. Parker: The Effect of action recognition and robot awareness in cooperative robotic teams, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IEEE, Pittsburgh 1995)* pp. 212–219
- 40.22 M. Mataríć: Behavior-based control: Examples from navigation, learning, and group behavior, *J. Exp. Theor. Artif. Intell.* **19**(2–3), 323–336 (1997)
- 40.23 B. MacLennan, G.M. Burghardt: Synthetic ethology and the evolution of cooperative communication, *Adapt. Behav.* **2**, 161–188 (1993)
- 40.24 T. Balch, R.C. Arkin: Communiation in reactive multiagent robotic systems, *Auton. Robot.* **1**(1), 27–52 (1995)
- 40.25 G. Dudek, M. Jenkin, E. Milios, D. Wilkes: A taxonomy for multi-agent robotics, *Auton. Robot.* **3**, 375–397 (1996)
- 40.26 G. Theraulaz, S. Goss, J. Gervet, J.-L. Deneubourg: Task differentiation in *Polistes* wasp colonies: A model for self-organizing groups of robots, *Proc. 1st Int. Conf. Simul. Adaptive Behavior* (Paris 1990) pp. 346–355
- 40.27 L. Steels: *Cooperation Between Distributed Agents Through Self-Organization*, ed. by Y. Demazeau, J.-P. Muller (Elsevier Science, Amsterdam 1990)

- 40.28 A. Drogoul, J. Ferber: From tom thumb to the dockers: Some experiments with foraging robots, *Proc. 2nd Int. Conf. Simul. Adaptive Behavior* (Honolulu 1992) pp. 451–459
- 40.29 M.J. Mataric: Designing emergent behaviors: From local interactions to collective intelligence, *Proc. 2nd Int. Conf. Simul. Adaptive Behavior*, ed. by J. Meyer, H. Roitblat, S. Wilson (MIT Press, Cambridge 1992) pp. 432–441
- 40.30 G. Beni, J. Wang: Swarm intelligence in cellular robotics systems, *Proc. NATO Adv. Workshop Robot. Biol. Syst.* (1989)
- 40.31 D. Stilwell, J. Bay: Toward the development of a material transport system using swarms of ant-like robots, *Proc. IEEE Int. Conf. Robot. Autom.* (Atlanta 1993) pp. 766–771
- 40.32 T. Fukuda, S. Nakagawa, Y. Kawauchi, M. Buss: Self organizing robots based on cell structures – CEBOT, *Proc IEEE Int. Workshop Intell. Robot. Syst.* (IEEE, 1988) pp. 145–150
- 40.33 J.H. Reif, H.Y. Wang: Social Potential fields: A distributed behavior control for autonomous robots, *Robot. Auton. Syst.* **27**(3), 171–194 (1999)
- 40.34 L.E. Parker: Designing control laws for cooperative agent teams, *Proc. IEEE Robot. Autom. Conf.* (IEEE, Atlanta 1993) pp. 582–587
- 40.35 K. Sugihara, I. Suzuki: Distributed algorithms for formation of geometric patterns with many mobile robots, *J. Robot. Syst.* **13**(3), 127–139 (1996)
- 40.36 V. Gazi: Swarm aggregations using artificial potentials and sliding-mode control, *IEEE Trans. Robot.* **21**(6), 1208–1214 (2005)
- 40.37 J. McLurkin, J. Smith: Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots, *Symp. Distrib. Auton. Robot. Syst.* (Springer, 2004)
- 40.38 D. Gage: Randomized search strategies with imperfect sensors, *Proc. SPIE Mobile Robots VIII* (SPIE, Boston 1993) pp. 270–279
- 40.39 T. Balch: The impact of diversity on performance in robot foraging, *Proc. 3rd Ann. Conf. Auton. Agents* (ACM Press, Seattle 1999) pp. 92–99
- 40.40 A. Howard, M.J. Mataric, G.S. Sukhatme: An incremental self-deployment algorithm for mobile sensor networks, *Auton. Robot.* **13**(2), 113–126 (2002), Special Issue on Intelligent Embedded Systems
- 40.41 Z.J. Butler, A.A. Rizzi, R.L. Hollis: Cooperative coverage of rectilinear environments, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, San Francisco 2000)
- 40.42 A.I. Mourikis, S.I. Roumeliotis: Performance analysis of multirobot cooperative localization, *IEEE Trans. Robot.* **22**(4), 666–681 (2006)
- 40.43 R. Grabowski, L.E. Navarro-Serment, C.J. Paredis, P.K. Khosla: Heterogeneous teams of modular robots for mapping and exploration, *Auton. Robot.* **8**(3), 271–298 (2000)
- 40.44 M. Berhault, H. Huang, P. Keskinocak, S. Koenig, W. Elmaghraby, P. Griffin, A. Kleywegt: Robot exploration with combinatorial auctions, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.* (IEEE, 2003) pp. 1957–1962
- 40.45 J. Kim, J.M. Esposito, V. Kumar: An RRT-based algorithm for testing and validating multi-robot controllers, *Proc. Robot.: Sci. Syst. I* (2005)
- 40.46 Z. Cao, M. Tin, L. Li, N. Gu, S. Wang: Cooperative hunting by distributed mobile robots based on local interaction, *IEEE Trans. Robot.* **22**(2), 403–407 (2006)
- 40.47 R.W. Beard, T.W. McLain, M. Goodrich: Coordinated target assignment and intercept for unmanned air vehicles, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Washington 2002)
- 40.48 J. Clark, R. Fierro: Cooperative hybrid control of robotic sensors for perimeter detection and tracking. (IEEE) pp. 3500–3505
- 40.49 P. Stone, M. Veloso: A layered approach to learning client behaviors in the RoboCup soccer server, *Appl. Artif. Intell.* **12**, 165–188 (1998)
- 40.50 J. McLurkin: Stupid Robot Tricks: Behavior-Based Distributed Algorithm Library for Programming Swarms of Robots. M.S. Thesis (Massachusetts Institute of Technology, Cambridge 2004)
- 40.51 L.E. Parker: The Effect of Heterogeneity in Teams of 100+ Mobile Robots. In: *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L.E. Parker, F. Schneider (Kluwer, Dordrecht 2003)
- 40.52 T. Balch: Hierarchic social entropy: An information theoretic measure of robot team diversity, *Auton. Robot.* **8**(3), 209–238 (2000)
- 40.53 D. Jung, A. Zelinsky: Grounded symbolic communication between heterogeneous cooperating robots, *Auton. Robot.* **8**(3), 269–292 (2000)
- 40.54 R.R. Murphy: Marsupial robots for urban search and rescue, *IEEE Intell. Syst.* **15**(2), 14–19 (2000)
- 40.55 G. Sukhatme, J. F. Montgomery, R. T. Vaughan: *Experiments with Cooperative Aerial-Ground Robots, Robot Teams: From Diversity to Polymorphism*, ed. by T. Balch, L. E. Parker (A K Peters, Natick 2002) pp. 345–368
- 40.56 L.E. Parker, B. Kannan, F. Tang, M. Bailey: Tightly-coupled navigation assistance in heterogeneous multi-robot teams, *Proc. IEEE Int. Conf. Intell. Robot. Syst.* (IEEE, 2004)
- 40.57 A. Howard, L.E. Parker, G.S. Sukhatme: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment, and detection, *Int. J. Robot. Res.* **25**, 431–447 (2006)
- 40.58 L. Chaimowicz, B. Grocholsky, J.F. Keller, V. Kumar, C.J. Taylor: Experiments in multirobot air-ground coordination, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, New Orleans 2004)
- 40.59 L.E. Parker, F. Tang: Building multi-robot coalitions through automated task solution synthesis, *Proc.*

- IEEE **94**(7), 1289–1305 (2006), special issue on Multi-Robot Systems
- 40.60 R. Zlot, A. Stentz: Market-based multirobot coordination for complex tasks, *Int. J. Robot. Res.* **25**(1), 73–101 (2006)
- 40.61 J. Jennings, C. Kirkwood-Watts: Distributed mobile robotics by the method of dynamic teams, *Proc. 4th Int. Symp. Distrib. Auton. Robot. Syst. Karlsruhe 1998*, ed. by T. Leuth, R. Dillman, P. Dario, H. Worn (Springer, Tokyo 1998)
- 40.62 E. Pagello, A. D'Angelo, E. Menegatti: Cooperation issues and distributed sensing for multirobot systems, *Proc. IEEE* **94**, 1370–1383 (2006)
- 40.63 B. Gerkey, M.J. Mataric: A formal analysis and taxonomy of task allocation in multi-robot systems, *Int. J. Robot. Res.* **23**(9), 939–954 (2004)
- 40.64 D. Gale: *The Theory of Linear Economic Models* (McGraw-Hill, New York 1960)
- 40.65 E. Balas, M.W. Padberg: On the set-covering problem, *Oper. Res.* **20**(6), 1152–1161 (1972)
- 40.66 B.B. Werger, M.J. Mataric: Broadcast of local eligibility for multi-target observation. In: *Distributed Autonomous Robotic Systems 4*, ed. by L.E. Parker, G. Bekey, J. Barhen (Springer, Tokyo 2000) pp. 347–356
- 40.67 R.A. Brooks: A robust layered control system for a mobile robot, *IEEE J. Robot. Autom.* **RA-2**(1), 14–23 (1986)
- 40.68 S. Botelho, R. Alami: M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Detroit 1999) pp. 1234–1239
- 40.69 R.G. Smith: The contract net protocol: High-level communication and control in a distributed problem solver, *IEEE Trans. Comput.* **C-29**(12), 1104–1113 (1980)
- 40.70 B. Dias, R. Zlot, N. Kalra, A. Stentz: Market-based multirobot coordination: A survey and analysis, *Proc. IEEE* **94**(7), 1257–1270 (2006)
- 40.71 B.P. Gerkey, M.J. Mataric: Sold! Auction methods for multi-robot coordination, *IEEE Trans. Robot. Autom.* **18**(5), 758–768 (2002)
- 40.72 H. Kose, U. Tatlidede, C. Mericli, K. Kaplan, H. L.Akin: Q-learning based market-driven multi-agent collaboration in robot soccer, *Proc. Turkish Symp. Artif. Intell. Neural Networks* (Izmir 2004) pp. 219–228
- 40.73 D. Vail, M. Veloso: Multi-robot dynamic role assignment and coordination through shared potential fields, multi-robot systems: From swarms to intelligent automata, *Proc. Int. Workshop Multi-Robot Syst.*, Washington, D.C., ed. by A. Schultz, L.E. Parker, F. Schneider (Springer, Dordrecht 2003) pp. 87–98
- 40.74 M. Lagoudakis, E. Markakis, D. Kempe, P. Keshinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, S. Jain: *Auction-based multi-robot routing, Robotics: Science and Systems I* (MIT Press, Cambridge 2005)
- 40.75 G. Rabideau, T. Estlin, S. Schien, A. Barrett: A comparison of coordinated planning methods for cooperating rovers, *Proc. AIAA Space Technol. Conf.* (1999)
- 40.76 R. Zlot, A. Stentz, M.B. Dias, S. Thayer: Multi-robot exploration controlled by a market economy, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Washington 2002) pp. 3016–3023
- 40.77 J. Guerrero, G. Oliver: Multi-robot task allocation strategies using auction-like mechanisms, *Proc. 6th Congr. Catalan Assoc. Artif. Intell.* (2003) pp. 111–122
- 40.78 N. Kalra, D. Ferguson, A. Stentz: Hoplites: A market-based framework for planned tight coordination in multirobot teams, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Barcelona 2005)
- 40.79 L. Lin, Z. Zheng: Combinatorial bids based multi-robot task allocation method, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Barcelona 2005) pp. 1145–1150
- 40.80 C.-H. Fua, S.S. Ge: COBOS: Cooperative backoff adaptive scheme for multirobot task allocation, *IEEE Trans. Robot.* **21**(6), 1168–1178 (2005)
- 40.81 E.G. Jones, B. Browning, M.B. Dias, B. Argall, M. Veloso, A. Stentz: Dynamically formed heterogeneous robot teams performing tightly-coupled tasks, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Orlando 2006) pp. 570–575
- 40.82 L. Vig, J.A. Adams: Multi-robot coalition formation, *IEEE Trans. Robot.* **22**(4), 637–649 (2006)
- 40.83 M. Bowling, M. Veloso: Simultaneous adversarial multi-robot learning, *Proc. Int. Joint Conf. Artif. Intell.* (2003)
- 40.84 F. Fernandez, L.E. Parker: A reinforcement learning algorithm in cooperative multi-robot domains, *J. Intell. Robot. Syst.* **43**, 161–174 (2005)
- 40.85 C.F. Touzet: Robot awareness in cooperative mobile robot learning, *Auton. Robot.* **2**, 1–13 (2000)
- 40.86 R. Steeb, S. Cammarata, F. Hayes-Roth, P. Thorndyke, R. Wesson: *Distributed Intelligence for Air Fleet Control*, Rand Corp. Technical Report, Number R-2728-AFPA (1981)
- 40.87 M. Benda, V. Jagannathan, R. Dodhiawalla: On optimal cooperation of knowledge sources, *Boeing AI Center Technical Report BCS-G2010-28* (1985)
- 40.88 T. Haynes, S. Sen: *Evolving Behavioral Strategies in Predators and Prey, Adaptation and Learning in Multi-Agent Systems*, ed. by G. Weiss, S. Sen (Springer, Berlin, Heidelberg 1986) pp. 113–126
- 40.89 S. Mahadevan, J. Connell: Automatic programming of behavior-based robots using reinforcement learning, *Proc. AAAI* (1991) pp. 8–14
- 40.90 S. Marsella, J. Adibi, Y. Al-Onaizan, G. Kaminka, I. Muslea, M. Tambe: On being a teammate: Experiences acquired in the design of RoboCup teams,

- Proc. 3rd Ann. Conf. Auton. Agents, ed. by O. Etzioni, J. Muller, J. Bradshaw (1999) pp. 221–227
- 40.91 J. Pugh, A. Martinoli: Multi-robot learning with particle swarm optimization, Proc. 5th Int. Joint Conf. Auton. Agents Multiagent Syst., Hakodate (ACM, New York 2006) pp. 441–448
- 40.92 P. Stone, M. Veloso: Multiagent systems: A survey from a machine learning perspective, Auton. Robot. **8**(3), 345–383 (2000)
- 40.93 M. Asada, E. Uchibe, S. Noda, S. Tawaratsumida, K. Hosoda: Coordination of multiple behaviors acquired by a vision-based reinforcement learning, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robot. Syst. (Munich 1994) pp. 917–924
- 40.94 M. Kubo, Y. Kakazu: Learning coordinated motions in a competition for food between ant colonies, Proc. 3rd Int. Conf. Simul. Adaptive Behavior, ed. by D. Cliff, P. Husbands, J.-A. Meyer, S. Wilson (MIT Press, Cambridge 1994) pp. 487–492
- 40.95 R. Alami, S. Fleury, M. Herrb, F. Ingrand, F. Robert: Multi-robot cooperation in the MARTHA project, Robot. Autom. Mag. **5**(1), 36–47 (1998)
- 40.96 A. Stroupe, A. Okon, M. Robinson, T. Huntsberger, H. Aghazarian, E. Baumgartner: Sustainable cooperative robotic technologies for human and robotic outpost infrastructure construction and maintenance, Auton. Robot. **20**(2), 113–123 (2006)
- 40.97 H.R. Everett, R.T. Laird, D.M. Carroll, G.A. Gilbreath, T.A. Heath-Pastore, R.S. Inderieden, T. Tran, K.J. Grant, D.M. Jaffee: Multiple Resource Host Architecture (MRHA) for the Mobile Detection Assessment Response System (MDARS), SPAWAR Systems Technical Document 3026, Revision A (2000)
- 40.98 Y. Guo, L.E. Parker, R. Madhavan: Towards collaborative robots for infrastructure security applications, Proc. Int. Symp. Collab. Technol. Syst. (2004) pp. 235–240
- 40.99 C. Hazard, P.R. Wurman, R. D’Andrea: Alphabet Soup: A testbed for studying resource allocation in multi-vehicle systems, Proc. AAAI Workshop Auction Mechan. Robot Coord. (AAAI, Boston 2006) pp. 23–30
- 40.100 D. Nardi, A. Farinelli, L. Iocchi: Multirobot systems: A classification focused on coordination, IEEE Trans. Syst. Man Cybernet. Part B **34**(5), 2015–2028 (2004)
- 40.101 K. Passino: Biomimicry of bacterial foraging for distributed optimization and control, IEEE Contr. Syst. Mag. **22**(3), 52–67 (2002)
- 40.102 M. Fontan, M. Mataric: Territorial multi-robot task division, IEEE Trans. Robot. Autom. **15**(5), 815–822 (1998)
- 40.103 I. Wagner, M. Lindenbaum, A.M. Bruckstein: Mac vs. PC – Determinism and randomness as complementary approaches to robotic exploration of continuous unknown domains, Int. J. Robot. Res. **19**(1), 12–31 (2000)
- 40.104 K. Sugawara, M. Sano: Cooperative behavior of interacting simple robots in a clockface arranged foraging field. In: *Distributed Autonomous Robotic Systems*, ed. by H. Asama, T. Arai, T. Fukuda, T. Hasegawa (Springer, Berlin, Heidelberg 2002) pp. 331–339
- 40.105 P. Rybski, S. Stoeter, C. Wyman, M. Gini: A cooperative multi-robot approach to the mapping and exploration of Mars, Proc. AAAI/IAAI–97 (AAAI, Providence 1997)
- 40.106 S. Sun, D. Lee, K. Sim: Artificial immune-based swarm behaviors of distributed autonomous robotic systems, Proc. IEEE Int. Conf. Robot. Autom. (IEEE, 2001) pp. 3993–3998
- 40.107 A.I. Mourikis, S.I. Roumeliotis: Optimal sensor scheduling for resource-constrained localization of mobile robot formations, IEEE Trans. Robot. **22**(5), 917–931 (2006)
- 40.108 S. Kloder, S. Hutchinson: Path planning for permutation-invariant multirobot formations, IEEE Trans. Robot. **22**(4), 650–665 (2006)
- 40.109 C.W. Reynolds: Flocks, herds and schools: A distributed behavioral model, ACM SIGGRAPH Computer Graphics **21**, 25–34 (1987)
- 40.110 T. Balch, R. Arkin: Behavior-based Formation Control for Multi-robot Teams, IEEE Trans. Robot. Autom. **14**(6), 926–939 (1998)
- 40.111 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Autom. Contr. **48**(6), 988–1001 (2002)
- 40.112 C. Belta, V. Kumar: Abstraction and control for groups of robots, IEEE Trans. Robot. **20**(5), 865–875 (2004)
- 40.113 C.M. Topaz, A.L. Bertozzi: Swarming patterns in two-dimensional kinematic model for biological groups, SIAM J. Appl. Math. **65**(1), 152–174 (2004)
- 40.114 J.A. Fax, R.M. Murray: Information flow and cooperative control of vehicle formations, IEEE Trans. Autom. Contr. **49**(9), 1465–1476 (2004)
- 40.115 J.A. Marshall, M.E. Broucke, B.R. Francis: Formations of vehicles in cyclic pursuit, IEEE Trans. Autom. Contr. **49**(11), 1963–1974 (2004)
- 40.116 S.S. Ge, C.-H. Fua: Queues and artificial potential trenches for multirobot formations, IEEE Trans. Robot. **21**(4), 646–656 (2005)
- 40.117 P. Tabuada, G. Pappas, P. Lima: Motion feasibility of multi-agent formations, IEEE Trans. Robot. **21**(3), 387–392 (2005)
- 40.118 G. Antonelli, S. Chiaverini: Kinematic control of platoons of autonomous vehicles, IEEE Trans. Robot. **22**(6), 1285–1292 (2006)
- 40.119 J. Fredslund, M.J. Mataric: A general algorithm for robot formations using local sensing and minimal communication, IEEE Trans. Robot. Autom. **18**(5), 837–846 (2002)
- 40.120 L.E. Parker: ALLIANCE: An architecture for fault tolerant, cooperative control of heterogeneous mobile robots, Proc. IEEE/RSJ/GI Int. Conf. Intell. Robot. Syst. (IEEE, Munich 1994) pp. 776–783

- 40.121 B. Donald, J. Jennings, D. Rus: Analyzing teams of cooperating mobile robots, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, 1994) pp. 1896–1903
- 40.122 S. Sen, M. Sekaran, J. Hale: Learning to coordinate without sharing information, *Proc. AAAI (AAAI, Seattle 1994)* pp. 426–431
- 40.123 B. Tung, L. Kleinrock: Distributed control methods, *Proc. 2nd Int. Symp. High Perform. Distrib. Comput.* (1993) pp. 206–215
- 40.124 Z.-D. Wang, E. Nakano, T. Matsukawa: Cooperating multiple behavior-based robots for object manipulation. (IEEE, Munich) pp. 1524–1531
- 40.125 P.J. Johnson, J.S. Bay: Distributed control of simulated autonomous mobile robot collectives in payload transportation, *Auton. Robot.* **2**(1), 43–63 (1995)
- 40.126 D. Rus, B. Donald, J. Jennings: Moving furniture with teams of autonomous robots, *Proc. IEEE/RJS Int. Conf. Intell. Robot. Syst.* (Pittsburgh 1995) pp. 235–242
- 40.127 F. Hara, Y. Yasui, T. Aritake: A kinematic analysis of locomotive cooperation for two mobile robots along a general wavy road, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Nagoya 1995) pp. 1197–1204
- 40.128 J. Sasaki, J. Ota, E. Yoshida, D. Kurabayashi, T. Arai: Cooperating grasping of a large object by multiple mobile robots, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Nagoya 1995) pp. 1205–1210
- 40.129 C. Jones, M.J. Mataric: Automatic synthesis of communication-based coordinated multi-robot systems, *Proc. IEEE/RJS Int. Conf. Intell. Robot. Syst.* (IEEE, Sendai 2004) pp. 381–387
- 40.130 Z. Wang, V. Kumar: Object closure and manipulation by multiple cooperating mobile robots, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, 2002) pp. 394–399
- 40.131 L.E. Parker: Cooperative robotics for multi-target observation, *Intell. Autom. Soft Comput.* **5**(1), 5–19 (1999)
- 40.132 S. Luke, K. Sullivan, L. Panait, G. Balan: Tunably decentralized algorithms for cooperative target observation, *Proc. 4th Int. Joint Conf. Auton. Agents Multiagent Syst., The Netherlands (ACM, New York 2005)* pp. 911–917
- 40.133 S.M. LaValle, H.H. Gonzalez-Banos, C. Becker, J.-C. Latombe: Motion strategies for maintaining visibility of a moving target, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Albuquerque 1997) pp. 731–736
- 40.134 A. Kolling, S. Carpin: Multirobot cooperation for surveillance of multiple moving targets – a new behavioral approach, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE, Orlando 2006) pp. 1311–1316
- 40.135 B. Jung, G. Sukhatme: Tracking targets using multiple mobile robots: The effect of environment occlusion, *Auton. Robot.* **13**(3), 191–205 (2002)
- 40.136 Z. Tang, U. Ozguner: Motion planning for multi-target surveillance with mobile sensor agents, *IEEE Trans. Robot.* **21**(5), 898–908 (2005)
- 40.137 D. Grossman: Traffic control of multiple robot vehicles, *IEEE J. Robot. Autom.* **4**, 491–497 (1988)
- 40.138 P. Caloud, W. Choi, J.-C. Latombe, C. Le Pape, M. Yim: Indoor automation with many mobile robots, *Proc. IEEE Int. Workshop Intell. Robot. Syst.* (IEEE, Tsuchiura 1990) pp. 67–72
- 40.139 H. Asama, K. Ozaki, H. Itakura, A. Matsumoto, Y. Ishida, I. Endo: Collision avoidance among multiple mobile robots based on rules and communication, *Proc. IEEE/RJS Int. Conf. Intell. Robot. Syst.* (IEEE, 1991)
- 40.140 S. Yuta, S. Premvuti: Coordinating autonomous and centralized decision making to achieve cooperative behaviors between multiple mobile robots, *Proc. IEEE/RJS Int. Conf. Intell. Robot. Syst.* (IEEE, Raleigh 1992) pp. 1566–1574
- 40.141 J. Wang: Fully distributed traffic control strategies for many-AGV systems, *Proc. IEEE Int. Workshop Intell. Robot. Syst.* (IEEE, 1991) pp. 1199–1204
- 40.142 J. Wang, G. Beni: Distributed computing problems in cellular robotic systems, *Proc. IEEE Int. Workshop Intell. Robot. Syst.* (IEEE, Tsuchiura 1990) pp. 819–826
- 40.143 H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, E. Osawa, H. Matasubara: RoboCup: A challenge problem of AI, *AI Mag.* **18**(1), 73–86 (1997)
- 40.144 H. Kitano, S. Tadokoro: RoboCup rescue: A grand challenge for multiagent and intelligent systems, *AI Mag.* **22**(1), 39–52 (2001)
- 40.145 B. Browning, J. Bruce, M. Bowling, M. Veloso: STP: Skills, tactics and plays for multi-robot control in adversarial environments, *IEEE J. Contr. Syst. Eng.* **219**, 33–52 (2005)
- 40.146 M. Veloso, P. Stone, K. Han: The CMUnited-97 robotic soccer team: Perception and multiagent control, *Robot. Auton. Syst.* **29**(2–3), 133–143 (1999)
- 40.147 T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, B. Nebel: CS Freiburg: coordinating robots for successful soccer playing, *IEEE Trans. Robot. Autom.* **5**(18), 685–699 (2002)
- 40.148 P. Stone, M. Veloso: Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, *Artif. Intell.* **110**(2), 241–273 (1999)
- 40.149 C. Candea, H.S. Hu, L. Iocchi, D. Nardi, M. Piaggio: Coordination in multi-agent Robocup teams, *Robot. Auton. Syst.* **36**(2), 67–86 (2001)
- 40.150 L.E. Parker: Current research in multirobot teams, *Artif. Life Robot.* **7**(2–3), 1–5 (2005)
- 40.151 K.R. Baghaei, A. Agah: Task allocation and communication methodologies for multi-robot systems, *Intell. Autom. Soft Comput.* **9**, 217–226 (2003)
- 40.152 T. Balch, L.E. Parker: Guest editorial, special issue on heterogeneous multi-robot systems, *Auton. Robot.* **8**(3), 207–208 (2000)

- 40.153 M. Dorigo, E. Sahin: Guest editorial, special issue on swarm robotics, *Auton. Robot.* **17**(2-3), 111-113 (2004)
- 40.154 M. Veloso, D. Nardi: Special issue on multirobot systems, *Proc. IEEE* **94**, 1253-1256 (2006)
- 40.155 T. Balch: Taxonomies of multi-robot task and reward. In: *Robot Teams: From Diversity to Polymorphism*, ed. by T. Balch, L.E. Parker (A K Peters, Natick 2002)
- 40.156 L.E. Parker, G. Bekey, J. Barhen (Eds.): *Distributed Autonomous Robotic Systems 4* (Springer, Berlin, Heidelberg 2000)
- 40.157 H. Asama, T. Arai, T. Fukuda, T. Hasegawa (Eds.): *Distributed Autonomous Robotic Systems 5* (Springer, Berlin, Heidelberg 2002)
- 40.158 A. Schultz, L.E. Parker (Eds.): *Multi-Robot Systems: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2002)
- 40.159 A. Schultz, L.E. Parker, F. Schneider (Eds.): *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2003)
- 40.160 E. Sahin, W.M. Spears (Eds.): *Swarm Robotics: SAB 2004 International Workshop* (Springer, Berlin, Heidelberg 2004)
- 40.161 L.E. Parker, F. Schneider, A. Schultz (Eds.): *Multi-Robot Systems Volume III: From Swarms to Intelligent Automata* (Kluwer, Dordrecht 2005)
- 40.162 R. Alami, R. Chatila, H. Asama (Eds.): *Distributed Autonomous Robotic Systems 6* (Springer, Berlin, Heidelberg 2006)
- 40.163 M. Gini, R. Voyles (Eds.): *Distributed Autonomous Robotic Systems 7* (Springer, Berlin, Heidelberg 2006)
- 40.164 T. Balch, L.E. Parker (Eds.): *Robot Teams: From Polymorphism to Diversity* (A K Peters, Natick 2002)

Networked R

41. Networked Robots

Vijay Kumar, Daniela Rus, Gaurav S. Sukhatme

This chapter discusses *networked robots*, multiple robots operating together coordinating and cooperating by *networked communication* to accomplish a specified task. This chapter presents an overview of the field with an emphasis on recent results and research challenges. Multiple robots enable new capabilities and the communication network enables new approaches and solutions that are difficult with just perception and control. Communication enables new control and perception capabilities in the system (e.g., access to information outside the perception range of the robot system). Conversely, control enables solutions for problems that are difficult without mobility (e.g., localization). Section 41.1 defines the field, examines the benefits of networking in robot coordination, and discusses applications. Section 41.2 highlights a few projects focused on networked robotics and discusses the application potential of the field. Section 41.3 discusses the research challenges at the intersection of control,

41.1 Overview.....	943
41.2 State of the Art and Potential	945
41.3 Research Challenges	947
41.4 Control	949
41.5 Communication for Control.....	950
41.6 Communication for Perception.....	951
41.7 Control for Perception	952
41.8 Control for Communication.....	953
41.9 Conclusions and Further Reading.....	955
References	955

communication, and perception. Section 41.4 defines a model for the control of a networked system which is used in Sects. 41.5–41.8 to examine specific research issues and opportunities facilitated by the interplay between communication, control, and perception.

41.1 Overview

The term *networked robots* refers to multiple robots operating together coordinating and cooperating by *networked communication* to accomplish a specified task. Communication between entities is fundamental to cooperation (and coordination), hence there is a central role for the communication network in networked robots. Networked robots may also involve coordination and cooperation with stationary sensors, embedded computers, and human users. The central feature of networked robots is the ability of the system to perform tasks that are well beyond the abilities of a single robot or multiple uncoordinated robots.

The *IEEE technical committee on networked robots* has adopted the following definition of a networked robot:

A networked robot is a robotic device connected to a communications network such as the Internet or local-area network (LAN). The network could be wired or wireless, and based on any of a variety of protocols such as the transmission control protocol (TCP), the user datagram protocol (UDP), or 802.11. Many new applications are now being developed ranging from automation to exploration. There are two subclasses of networked robots:

1. Teleoperated, where human supervisors send commands and receive feedback via the network. Such systems support research, education, and public awareness by making valuable resources accessible to broad audiences.

2. Autonomous, where robots and sensors exchange data via the network. In such systems, the sensor network extends the effective sensing range of the robots, allowing them to communicate with each other over long distances to coordinate their activity. Sensing, actuation, and computation need no longer be collocated. A broad challenge is to develop a science base that couples communication, perception, and control to enable such new capabilities.

This definition of autonomous networked robots also includes a third class of distributed systems, mobile sensor networks, which is a natural evolution of sensor networks. Robot networks allow robots to measure spatially and temporally distributed phenomena more efficiently. The robots in turn can deploy, repair, and maintain the sensor network to increase its longevity, and utility. The focus of this chapter is *autonomous networked robots*.

Embedded computers and sensors are becoming ubiquitous in homes and factories, and increasingly wireless ad hoc networks or plug-and-play wired networks are becoming commonplace. Human users interact with embedded computers and sensors to perform tasks ranging from monitoring (e.g., supervising the operation of a factor and surveillance in a building) to control (e.g., running an assembly line consisting of sensors, actuators, and material-handling equipment). In most of these cases, the human users, embedded comput-

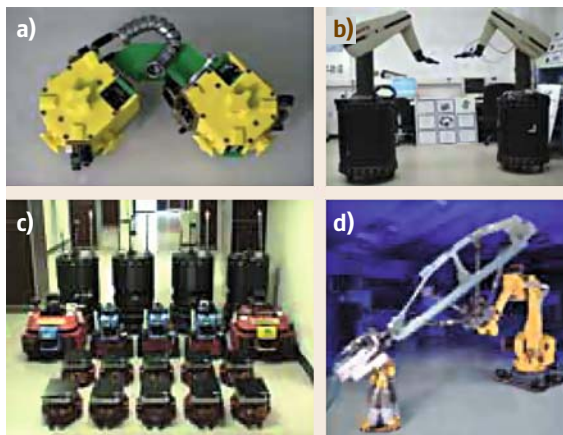


Fig. 41.1a–d Small modules [41.1] can automatically connect and communicate information to perform locomotion tasks (a); robot arms [41.2] on mobile bases can cooperate to perform household chores (b); swarms of robots [41.3] can be used to explore an unknown environment (c); and industrial robots can cooperate in welding operations (d)

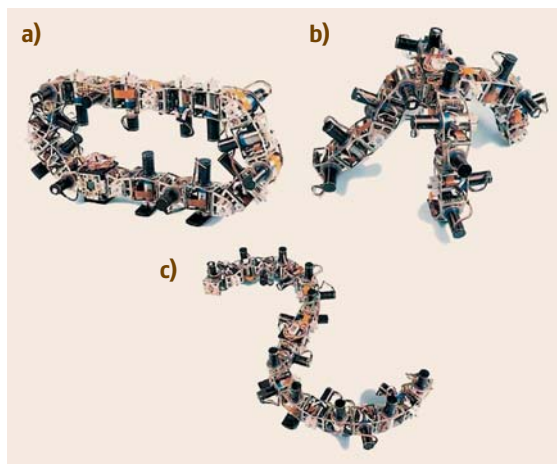


Fig. 41.2a–c Robotic modules [41.4] can be reconfigured to morph into different locomotion systems including a wheel-like rolling system (a), a snake-like undulatory locomotion system (b), a four-legged walking system (c)

ers, and sensors are not collocated and the coordination and communication happens through a network. *Networked robots* extends this vision to multiple robots functioning in a wide range of environments performing tasks that require them to coordinate with other robots, cooperate with humans, and act on information derived from multiple sensors.

Figure 41.1 shows prototype concepts derived from academic laboratories and industry. In all these examples, independent robot or robotic modules can cooperate to perform tasks that a single robot (or module) cannot perform. Robots can automatically couple to perform locomotion tasks (also see Fig. 41.2) and manipulation tasks that either a single robot cannot perform, or that would require a special-purpose larger robot to perform. They can also coordinate to perform search and reconnaissance tasks exploiting the efficiency that is inherent in parallelism. They can also perform independent tasks that need to be coordinated. Examples in the manufacturing industry include, for example, fixturing and welding.

Besides being able to perform tasks that individual robots cannot perform, networked robots also result in improved efficiency. Networking gives each robot access to information outside its perception range. Tasks such as searching or mapping can, in principle, be performed faster with an increase in the number of robots. A speed up in manufacturing operations can be achieved by deploying multiple robots performing operations in parallel but in a coordinated fashion.

Another advantage of using the network to connect robots is the ability to connect and harness physically removed assets. Mobile robots can react to information sensed by other mobile robots at a remote location. Industrial robots can adapt their end-effectors to new parts being manufactured upstream in the assembly line. Human users can use machines that are remotely located via the network.

The ability to network robots also enables fault tolerance in design. If robots can dynamically reconfigure themselves using the network, they are more tolerant to robot failures. This is seen in the Internet where multiple gateways, routers, and computers provide for a fault-tolerant system (although the Internet is not robust in other ways). Similarly, robots that can *plug* and *play* can be swapped in and out automatically to provide for a robust operating environment.

Finally, networked robots have the potential to provide great synergy by bringing together components

with complementary benefits and making the whole greater than the sum of the parts.

Applications for networked robots abound. The US military routinely deploys unmanned vehicles that are reprogrammed remotely based on intelligence gathered by other unmanned vehicles, sometimes automatically. The deployment of satellites in space, often by astronauts in a shuttle with the shuttle robot arm requires the coordination of complex instrumentation onboard the space shuttle, human operators on a ground station, the shuttle arm, and a human user on the shuttle. Home appliances now contain sensors and are becoming networked. As domestic and personal robots become more commonplace, it is natural to see these robots working with sensors and appliances in the house while cooperating with one or more human users. Networked robots will likely be used as critical ingredients in the environmental observatories of the future. Large-scale ecological monitoring precludes the use of monolithic infrastructure, and is envisioned to be built as a distributed, networked robotic system.

41.2 State of the Art and Potential

The Network Robot Forum established in Japan in 2003 [41.5] estimates that the size of the networked robot industry will be over \$200B by 2013, much larger than the industrial robot market for manufacturing applications. This growth is broad-based, across many industries. There is a strong connection between this industry and the industry connected to sensor networks. Sensor networks have been projected to grow dramatically in terms of commercialization and market value [41.6]. Robot networks are analogous to sensor networks except that they allow sensors to have mobility and allow the geographical distribution of the sensors to be adapted based on the information acquired.

A system of robots, embedded computers, actuators, and sensors has tremendous potential in civilian, defense, and manufacturing applications. Nature provides the proof of concept of what is possible [41.7]. Group behaviors in nature can be found in organisms that are only microns to those that are several meters in length. There are numerous examples of simple animals that execute simple behaviors with modest sensors and actuators but communicate with and sense nearest neighbors to enable complex emergent behaviors that are fundamental to navigation, foraging, hunting, constructing nests, survival, and eventually growth. As seen

in Fig. 41.3, relatively small agents are able to manipulate objects that are significantly larger in terms of size and payload by cooperating with fairly simple individual behaviors. The coordination between agents is completely decentralized, allowing scaling up to large numbers of robots and large objects [41.8]. Individuals do not recognize each other. In other words, there is no labeling or identification of robots. The number



Fig. 41.3 Ants are able to cooperatively manipulate and transport objects often in large groups, without identified or labeled neighbors, and without centralized coordination

of agents in the team is not explicitly encoded. Agents are identical, enabling robustness to failures and modularity. There is minimal communication, and even that which is present is only between neighbors. Furthermore, the optimal mode of group coordination may be scale dependent. Studies of wasps show strong evidence of centralized coordination among species with small colony sizes, but a distributed, decentralized coordination in larger colonies [41.9]. All these attributes are relevant to networked robots.

Biology has shown how simple decentralized behaviors in unidentified individuals (e.g., insects and birds exhibiting swarming behaviors) can exhibit a wide array of seemingly intelligent group behaviors. Similarly networked robots can potentially communicate and cooperate with each other, and even though individual robots may not be sophisticated, it is possible for networked robots to provide a range of intelligent behaviors that are beyond the scope of intelligent robots.

The significance and potential impact of networked robots is apparent from the following examples.

The manufacturing industry has always relied on integration between sensors, actuators, material-handling equipment, and robots. Today companies are finding it easier to reconfigure existing infrastructure by networking new robots and sensors with existing robots via wireless networks. There is also an increasing trend toward robots interacting with each other in operations like welding and machining, and robots cooperating with humans in assembly and material-handling tasks. Workcells consist of multiple robots, numerous sensors and controllers, automated guided vehicles, and one or two human operators working in a supervisory role. However, in most of these cells, the networked robots operate in a structured environment with very little variation in configuration and/or operating conditions.

There is a growing emphasis on networking robots in applications of field robotics, for example, in the mining industry. Like the manufacturing industry, operating conditions are often unpleasant and the tasks are repetitive. However, these applications are less structured and human operators play a more important role.

In the health care industry, networks allow health care professionals to interact with their patients, other professionals, expensive diagnostic instruments, and in the future surgical robots. Telemedicine is expected to provide a major growth impetus for remote networked robotic devices that will take the place of today's stand-alone medical devices.

There are already many commercial products, notably in Japan, where robots can be programmed via

and communicate with cellular phones. For example, the MARON robot developed by Fujitsu lets a human user dial up their robot and instruct it to conduct simple tasks including sending pictures back to the user via a cellular phone. Indeed these robots will interact with other sensors and actuators in the home – door openers equipped with Bluetooth cards and actuators and computer-controlled lighting, microwaves, and dishwashers. Indeed the Network Robot Forum [41.5] is already setting standards for how stationary sensors and actuators can interact with other robots in domestic and commercial settings.

Environmental monitoring is a key application for networked robots. By exploiting mobility and communication, robotic infrastructure enables observation and data collection at unprecedented scales in various aspects of ecological monitoring. This is significant for environmental regulatory policies (e.g., clean air and water legislation), as well as an enabler of new scientific discovery, for example, it is possible to obtain maps of salinity gradients in oceans, temperature and humidity variations in forests, the and chemical composition of air and water in different ecological systems [41.10]. In addition to mobile sensor networks, it is also possible to use robots to deploy sensors and to retrieve information from the sensors. Mobile platforms allow the same sensor to collect data from multiple locations while communication allows the coordinated control and aggregation of information. Examples include systems built for aquatic [41.11], terrestrial [41.12], and subsoil monitoring [41.13]. There are many efforts to develop networked underwater platforms [41.14–16]. Networks of static and robotic devices have been developed for aquatic monitoring [41.11] and to obtain high-resolution information on the spatial and temporal distributions of plankton assemblages and concomitant environmental parameters. The RiverNet project [41.17] at Rensselaer Polytechnic Institute (RPI) has focused on the development of robotic sensor networks for monitoring a river ecosystem. Recent work at University of California, Los Angeles (UCLA), University of Southern California (USC), University of California, Riverside, and University of California, Merced on the networked infomechanical system project [41.12] has focused on the development of robotic networks for monitoring the forest canopy, with a view to providing data for modeling canopy and undercover growth. Networked robotic mini-rhizotrons [41.13] are being deployed in the forest to monitor root growth in the soil.

In the defense industry, countries like the USA have invested heavily in the concept of networked, geo-

graphically distributed assets. Unmanned aerial vehicles like the Predators are operated remotely. Information from sensors on the Predators triggers the deployment of other vehicles and weapon systems at a different remote location and allows commanders in a third location to control and command all these assets. The US military is engaged in the large Future Combat Systems initiative to develop network-centric approaches to deploying autonomous vehicles. The network-centric tactical paradigms for modern warfare have created networked robots for defense and homeland security. While networked robots are already in operation, current approaches are limited to human users commanding a single vehicle or sensor system. However, it takes many human operators (between 2–10 depending on the complexity of the system) to deploy complex systems like unmanned aerial vehicles. A Predator unmanned aerial vehicle (UAV) is operated from a tactical control station, which may be on an aircraft carrier, with a basic crew of 3–10 operators.

The eventual goal, however, is to enable a single human user to deploy networks of unmanned aerial, ground, surface, and underwater vehicles. There have been several recent demonstrations of multirobot systems exploring urban environments [41.20, 21] and the interiors of buildings [41.19, 22] to detect and track intruders, and transmit all of the above information to a remote operator. These examples show that it is possible to deploy networked robots using an off-the-shelf 802.11b wireless network and have the team be remotely tasked and monitored by a single operator. An example of a project with heterogeneous vehicles in an urban setting is shown in Fig. 41.4. An example of a project with heterogeneous vehicles in an indoor setting is shown in Fig. 41.5 wherein robots map an environment and deploy themselves to form a sensor network to detect intruders.

Many research projects are addressing group behaviors or collective intelligence by realizing swarming behaviors observed in nature. For example, the European Union (EU) has several EU-wide coordinated projects on collective intelligence or swarm intelligence. The I-Swarm project in Karlsruhe [41.23] and the Swarm-Bot project at Ecole Polytechnique Fédérale

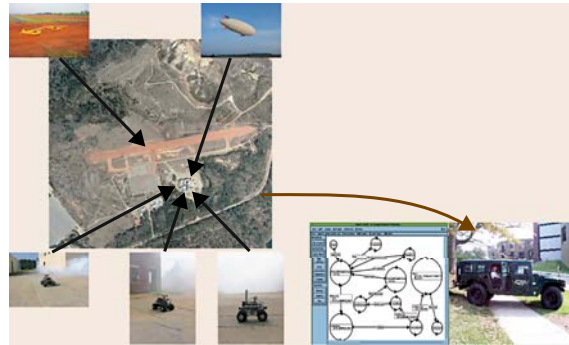


Fig. 41.4 A single operator commanding a network of aerial and ground vehicles from a command and control vehicle in an urban environment for scouting and reconnaissance in a recent demonstration by the University of Pennsylvania, Georgia Tech. and University of Southern California [41.18]



Fig. 41.5 Under the DARPA SDR program, a team from the University of Southern California, the University of Tennessee, and Science Applications and International Corporation (SAIC) demonstrated mapping, and intruder detection by a team of networked robots [41.19]

de Lausanne (EPFL) [41.24] are examples of swarm intelligence. The Laboratory for Analysis and Architecture of Systems (LAAS) has a strong group in robotics and artificial intelligence. This group has had a long history of basic and applied research in multirobot systems. The integration of multiple unmanned vehicles for applications such as terrain mapping and fire-fighting is addressed in [41.25]. A recent multi-university US project addresses the development of networked vehicles for swarming behaviors [41.26]. Projects such as these are exploring the scalability of the basic concepts to large numbers of robots, sensors, and actuators.

41.3 Research Challenges

While there are many successful embodiments of networked robots with applications to manufacturing industry, the defense industry, space exploration,

domestic assistance, and civilian infrastructure, there are significant challenges that have to be overcome.

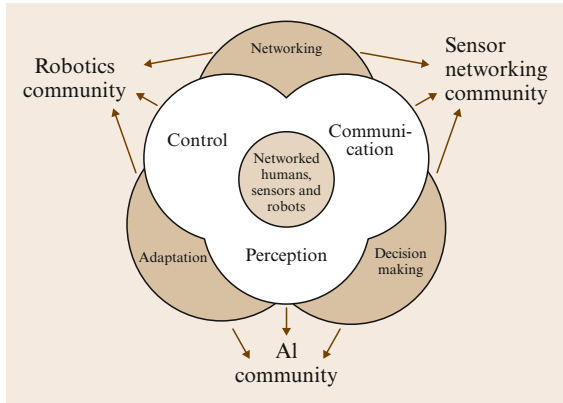


Fig. 41.6 The paradigm of *networked robots* introduces fundamental challenges at the intersection of control, perception, and communication that is of interest to the robotics, sensor networks, and artificial intelligence communities

The problem of coordinating multiple autonomous units and making them cooperate creates problems at the intersection of communication, control, and perception. Who should talk to who and what information should be conveyed, and how? How does each unit move in order to accomplish the task? How should the team members acquire information? How should the team aggregate information? These are all basic questions that need basic advances in control theory, perception, and networking. In addition, because humans are part of the network (as in the case of the Internet), we have to devise an effective way for multiple humans to be embedded in the network and command/control/monitor the network without worrying about the specificity of individual robots in the network. Thus the underlying research challenges lie at the intersection of control theory, perception, and communication/networks, as shown in Fig. 41.6.

It is also worth noting that robot networks are dynamic unlike networks of sensors, computers or machines which might be networked together in a fixed topology. When a robot moves, its neighbors change and its relationship to the environment changes. As a consequence, the information it acquires and the actions it executes must change. Not only is the network topology dynamic, but the robot's behavior also changes as the topology changes. It is very difficult to predict the performance of such dynamic robot networks, yet it is this analysis problem that designers of robot networks must solve before deploying the network.

This notion of a changing topology inevitably leads us to complicated mathematical models. Tradi-

tionally, models of group behavior have been built on continuous models of dynamics of individuals, including local interactions with neighbors, and models of control and sensing with a fixed set of neighbors. While dynamics at the level of individual units may be adequately described by differential equations, the interactions with neighbors are best described by edges on a graph. Modeling, analysis, and control of such systems will require a comprehensive theoretical framework and new representational tools. New mathematical tools that marry dynamical system theory, switched systems, discrete mathematics, graph theory, and computational geometry are needed to solve the underlying problems. We need a design methodology for solving the *inverse problem in navigation* – behaviors for controlling individuals to achieve a specified aggregate *motion* and *shape* of the group, and the application to active perception and coverage. An overview of some of these methods is provided in Sect. 41.4.

Problems of perception have been studied extensively in the robotics community. However, the perception problems in a system of networked, mobile sensor platforms bring a new set of challenges; for example, consider the problem of estimating the state of the network. State estimation requires the estimation of the state of robots and the environment based on local, limited-range sensory information. Localization of n vehicles in an m -dimensional configuration space requires $O((nm)^k)$ computations, where k is somewhere between 3 and 6, depending on the algorithm and domain-specific assumptions. The estimation problem is further exacerbated by the fact that not all robots in the network may be able to get the necessary information in a time-critical fashion. There are deep issues of representation and algorithmic development, which are discussed in Sect. 41.6.

The paradigm of active perception [41.27] links the control of sensor platforms to perception, bringing control theory and perception together in a common framework. Extending this paradigm to networked robots requires approaches of distributed control to be merged with decentralized estimation. Robots can move in order to localize themselves with respect to their neighbors, to localize their neighbors, and also to identify, localize, and track features in the environment. These problems are discussed in Sect. 41.7.

As discussed earlier, the communication network is central to the functioning of a network of robots. However, if the network consists of mobile agents with transmitters and receivers with finite power, there is no

guarantee that all agents can talk to teach other. Unlike a static sensor network, robots in a network can move toward each other to facilitate communication and adap-

tively maintain a communication network. Some basic algorithmic problems and several pertinent results are provided in Sect. 41.8.

41.4 Control

The control of individual robots is critical to the performance and scope of robot networks. Indeed motion coordination algorithms have been proposed for the purpose of improving communication performance [41.28, 29], localization [41.30, 31], information integration, deployment [41.32], and coverage [41.33–35], among other tasks. Mobility allows the group of robots to self-deploy, and self-organize by relocating themselves in support of communication, sensing, or task needs; for example, they can reconfigure to guarantee a desired communication bandwidth, k -hop connectivity, or algebraic connectivity, enabling message delivery from one robot to another. The group can also self-organize to position sensors so as to cover a desired area and adapt to shifts in the focus of monitoring activities. Controlling sensor position also supports map making, tracking of objects and events, and goal-directed navigation for users of the network. Finally, mobility allows robots to accomplish tasks such as navigation, reconnaissance, transportation, and search and rescue.

Given a group of mobile sensors, we would like to have distributed control capabilities that realize desirable global specifications. Thus, it is necessary to be able to automatically determine the necessary position and orientation of the group members and/or the distribution of group members, and their motion to achieve the desired task. At a lower level, the robots must be able to use information from the communication network and from their own sensors to derive local estimates, reason about the spatial network (their neighbors and their relationship to the environment), and then use the appropriate control policies to achieve the desired group specifications. We briefly outline the simplest mathematical model that is necessary to formulate such problems in order to provide a better sense of the underlying challenges.

In a robot network, we have multiple agents or nodes in which each agent is a physical entity that can be a robot, a vehicle with actuators and sensors, a sensor platform (possibly static) or even a communication relay node. Each agent A_i is characterized by an identifier, $i \in I \subset \mathbb{Z}$, a state $x_i \in X_i \subset \mathbb{R}^n$, and control inputs $u_i \in U_i \subset \mathbb{R}$, with $f_i : X_i \times U_i \rightarrow TX_i$ specifying

the dynamics:

$$\dot{x}_i = f_i(x_i, u_i). \quad (41.1)$$

The state x_i will consist of the position (and orientation), r_i in some d -dimensional space, and its velocity, $\dot{r}_i : x_i = (r_i^T, \dot{r}_i^T)^T$, with $n = 2d$. $\mathcal{N}^c(r_i)$ and $\mathcal{N}^s(r_i)$ are neighborhoods of r that define the range and field of view of the communication hardware and sensors, respectively.

A network of robots S consists of N agents with a *sensing graph* and a *communications graph* that is defined by the physical distribution of the agents. The sensing graph (and similarly the communications graph) is defined by a map $E^s : X^1 \times X^2 \dots X^N \rightarrow I \times I$, where the edges of the graph are formed dynamically depending on the physical proximity of pairs of agents. Specifically, the $N \times N$ adjacency matrix, \mathcal{A}^s (and similarly \mathcal{A}^c) has entries

$$\mathcal{A}_{ij}^s = \begin{cases} 1, & \text{if } r_j \in \mathcal{N}^s(r_i), \\ 0, & \text{otherwise.} \end{cases} \quad (41.2)$$

Agent A_i has estimates of its own state and the states of neighbors (e.g., A_j), and these estimates are derived from information associated with edges in the sensing and communication graph

$$\hat{x}_j^{(i)} = h(x_i, z_{ij}), \quad (41.3)$$

where z_{ij} represents measurements of the state of agent A_j available to A_i by sensing or communication channels and h is the estimator used by A_i . Note that z_{ij} may have dimension less than n and may therefore not contain complete information about $x_{ij} = x_i - x_j$. Clearly the relative position vector denoted by $r_{ij} = r_i - r_j$ and its magnitude are important quantities that may need to be estimated for biological and artificial agents.

Finally, A_i can encode n_{b_i} behaviors, which we will denote by $\mathcal{B}_i = B_1, B_2, \dots, B_{n_{b_i}}$. Each behavior B_j is a controller, a function $k_j : \mathbb{R} \times X_i \rightarrow U_i$. All agents can be assigned identical or different behaviors. Each behavior represents a set of unsynchronized, locally executed computations (for control or estimation) being carried out for some collective purpose, with each processor using in its computations only data from its *neighboring*

processors. Furthermore, even for a fixed assignment of behaviors, each processor's neighbors typically *change* with time because the processors are moving in and out of the sets \mathcal{N}^c and \mathcal{N}^s . Thus the methodology for modeling and analyzing such systems will require the merging of graph theory and dynamical system theory at a fundamental level.

The reader is directed to the many survey articles on this subject for further information. An overview

of challenges for the controls community is presented in [41.36]. The underlying theory for networked mobile systems has been explored in the context of automated highway systems [41.37], cooperative robot reconnaissance [41.19] and manipulation [41.38], formation flight control [41.39], and the control of groups of unmanned vehicles [41.21]. Our goal in the following sections is to explore the connections between communication, perception, and control.

41.5 Communication for Control

Communication networks allow physically disconnected entities to exchange information. At the lowest level, when groups of vehicles coordinate their actions, communication allows vehicles to exchange state information [41.40–42]. At a higher level, robots can plan navigation and exploration tasks based on an integrated map of the world derived from information acquired from different robots [41.43].

The use of communication for control in the multi-vehicle context has been addressed in the PATH project where formations of inline vehicles were studied [41.37]. Problems of the stability of the formation [41.44], the convergence of the formation to shapes [41.45], and the overall performance of the system [41.46] are of great interest. The performance of the system is directly influenced by the interconnections between agents. In addition to impacting on stability [41.37], feedback of states from different agents and feedforward information from the plans of different agents affects the rates at which the system of agents can respond to external stimuli [41.46] or to commands from human operators [41.47].

In addition, communication can be used for high-level control and planning of robots. There is great interest in using static sensor nodes as beacons to guide robot navigation. In [41.48], the problem of coverage and exploration of an unknown dynamic environment using a mobile robot is considered. An algorithm is presented which assumes that global information is not available [neither a map, nor global positioning system (GPS) information]. The algorithm deploys a network of radio beacons that assists the robot in coverage. The network is also used by the robot for navigation. The deployed network can also be used for applications other than coverage (such as multirobot task allocation). A similar idea was presented using potential-field-based

navigation in [41.43]. In this work the notion of no-go or danger areas was incorporated into the navigation cost function. Recent work along these lines with experimental data from sensor nodes is reported in [41.49].

In such communication-enabled cooperative control and planning (see also [41.50]), the communication network plays an important role in the creation of a shared representation of information. This notion of a shared representation is important to the scaling of coordinated control algorithms to large numbers of devices. For example, in [41.41], the information form of the Kalman filter is used to derive a framework for decentralizing estimation and fusion algorithms. This approach was shown to be applicable to multiple heterogeneous ground and aerial platforms [41.30]. Such methodologies are transparent to the specificity and identity of the cooperating vehicles. This is because vehicles share a common representation, which consists of a certainty grid that contains information about the probability of detection of targets, and an information vector–matrix pair that is used in the information form of the Kalman filter [41.21]. Observations are propagated through the network by changing both the certainty grid and the information vector/matrix. This allows each vehicle to choose the action that maximizes a utility function, which is the combined mutual information gain from onboard sensors towards the detection and localization of features in the environment.

Thus, in summary, at the lowest level, communication enables either partial or complete state feedback of the network and allows agents to exchange information for feedforward control. At the higher levels, agents can share information for planning and for control. This is also discussed in Sect. 41.6 where the communication network is shown to enable a network-centric approach to perception.

41.6 Communication for Perception

While individual robots have sensors and the ability to build maps and models by integrating sensory information, networked robots can exchange information and leverage sensory data, maps, and models from other robots. The challenge is to exploit communication for perception in tasks such as distributed mapping in the presence of the delays, limited bandwidth, and disruption that are typical of communication networks.

Distributed localization is the term used to describe the merging of communication and perception for state estimation. Localization is an essential tool for the development of low-cost robot networks for use in location-aware applications and ubiquitous networking [41.51]. Location information is needed to track the placement of the nodes and to correlate the values measured by the node with their physical location. Distributed computation and robustness in the presence of measurement noise are key ingredients for a practical localization algorithm that will give reliable results over a large-scale network.

The methods for distributed localization can be classified into two broad classes: algorithms that rely on anchor nodes for localization and algorithms that use no beacons. Localization may be computed using range information between nodes, bearing information, or both.

In [41.28] a theoretical foundation for network localization in terms of graph rigidity theory is provided. The problem is solved when nodes have perfect range information and it is shown that a network has a unique localization if and only if its underlying graph is *generically globally rigid*. In [41.52] the Cramér–Rao lower bound (CRLB) for network localization is derived. This work computes the expected error characteristics for an ideal algorithm, and compares this to the actual error in an algorithm based on multilateration, drawing the important conclusion that the error introduced by the algorithm is just as important as the measurement error in assessing end-to-end localization accuracy. In [41.53] a distributed algorithm that uses no beacons and is guaranteed to compute correct location information under measurement noise for nodes that can range to neighbors is presented. This algorithm relies on the notion of robust quadrilaterals to compute robustly a global system of coordinates among the nodes. The computation supports moving nodes. Extensions of this work to passive tracking have been discussed in [41.54]. Localization based on the propagation of location information

from known reference nodes based on connectivity includes [41.55,56]. Mobility-assisted localization is introduced in [41.57]. Other techniques use distributed propagation of location information using multilateration [41.52,58].

Two approaches for cooperative relative localization of mobile robot teams are given in [41.59,60]. Neither method uses GPS, landmarks, or maps of any kind; instead, robots make direct measurements of the relative pose of nearby robots and broadcast this information to the team as a whole. In [41.59], each robot processes this information independently to generate an egocentric estimate for the pose of other robots using a Bayesian formalism with a particle filter implementation. In [41.60], maximum-likelihood estimation (MLE) and numerical optimization is used to achieve a similar result.

A key issue is to be able to scale these computations for building a shared representation to large numbers of robots and sensors. This problem was studied in recent experiments under the US Defense Advanced Research Projects Agency (DARPA-funded software for distributed robotics (SDR) program. The goal of these experiments was to develop and demonstrate a multi-robot system capable of carrying out a specific mission. This required the ability to deploy a large number of robots into an unexplored building, map the building interior, detect and track intruders, and transmit all of the above information to a remote operator. A report on one set of experiments is presented in [41.19]. A tiered strategy for deploying the robots is described, where highly capable robots formed the first wave to enter and map a building, followed by a second wave which used the resulting map to self-deploy and monitor the environment for intruders. Both approaches relied extensively on networking the robots using commercial 802.11b wireless technology. This task involved both communication for building a shared representation as well control for perception.

Another important set of problems arises when robot networks are used for identifying, localizing, and then tracking targets in a dynamic setting. An embedded stationary wireless sensor network is like a virtual sensor spread over a large geographical area. Such a network can provide information to mobile robots about remote locations. Robot networks allow this virtual sensor to move in response to external stimuli and to track moving targets. Indeed, it is possible to cast this scenario as a pursuit-evasion game

with robotic sensor networks [41.61]. For example, the Tenet project at USC addresses the design of network primitives and abstractions for tiered network architectures, with robotic pursuit evasion as one of the target applications. Algorithms for guiding the sampling strategy of a robotic boat to model and locate phenomena of interest (e.g., hotspots) in aquatic environments are discussed in [41.11]. The networked infomechanical systems (NIMS) project has focused on sensor-assisted techniques for mobile robot-based adaptive sampling for event response [41.62] and field reconstruction [41.63].

The information collected by the nodes in a sensor network can be processed at a central location or in a decentralized fashion. Such in-network data processing techniques make better use of network communication and computation resources than centralized processing. This also enables the network to compute accurate and up-to-date global pictures of the global perception landscape that are available to all the robots in the system. Methods for in-network data processing with static nodes include artificial potential-field computation, gradient computations, particle filters, Bayesian inference, and signal processing. Algorithms

have been developed for computing maps, paths, and predictors [41.43, 48, 64].

A recent DARPA demonstration showed how communication networks can be used effectively in perception tasks involving heterogeneous robots [41.20]. In cooperative search, identification, and localization unmanned aerial vehicles (UAVs) can be used to cover large areas, searching for targets. However, sensors on UAVs are typically limited in their accuracy of localization of targets on the ground. On the other hand, ground robots can be deployed to accurately locate ground targets but have the disadvantage of not being able to move rapidly and see through obstacles such as buildings or fences. In [41.30], the synergy between these two devices is exploited by creating a seamless network of UAVs and unmanned ground vehicles (UGVs). As discussed in Sect. 41.5, the key to such network-centric approaches for search and localization is a shared representation of state information, which in this case is easily scalable to large numbers of UAVs and UGVs and is transparent to the specificity of individual platforms. However, how to do this more generally and for more unstructured information remains an issue for future research.

41.7 Control for Perception

Networked mobile robots enable the exploration of dynamic environments and the recovery of three-dimensional information via distributed active perception [41.27]. Since the nodes are mobile, a natural question is: where should the nodes be placed in order to ensure successful integration of information from multiple nodes, and to maximize the quality of the estimates returned by the team? Since there is a cost associated with transmitting and processing data, it is important to consider which sensor readings should be used in the state estimation and what information should be communicated to the rest of the system. The quality of the information computed by the network depends on the locations of the sensor platforms both in an absolute and relative sense. The quality also depends on the noise characteristics of each sensor, and the communication network.

A robot network goes well beyond a fixed sensor network, which can only collect data at fixed positions in space; for example, when an event is detected at a specific location it is possible to direct more sensors toward the location of observation of the event for more infor-

mation (for example, higher-resolution data or higher sampling frequency). Reconfiguring the node locations for adaptive resolution sampling relies on distributed control strategies.

Various strategies have been introduced for controlling mobile sensor network coverage. Mobile sensing agents are controlled using gradients of information-based objective functions [41.65]. Stability results are derived without concerns for the optimality of the network configuration, but local guarantees are provided. A body of results reported in [41.66] and [41.67] describes decentralized control laws for positioning mobile sensor networks optimally with respect to a known event distribution density function. This approach is advantageous because it guarantees that the network (locally) minimizes a cost function relevant to the coverage problem. However, the control strategy requires that each agent have a complete knowledge of the event distribution density, thus it is not reactive to the sensed environment. The work by [41.68, 69] generalizes these results to situations in which the nodes estimate rather than know ahead of time the event distribution density

function. A local (decentralized) control law requires that each agent can measure the value and gradient of the distribution density function at its own position. This results in a sensor network that is reactive to its sensed environment while maintaining or seeking a near-optimal sensing configuration. In addition, the distribution density function approximation yields a closed-form expression for the control law in terms of the vertices of an agent's Voronoi region. This eliminates the need for the numerical integration of a function over a polynomial domain at every time step, thereby providing a significant reduction in computational overhead for each agent. Other work in event monitoring for unknown distributions includes [41.33]. Krause et al. [41.70] have recently proposed an approach for sensor placement that considers both the sensing quality and communication cost of imperfect sensing and communication components. They use a parametric model for link reception rate that assumes no acknowledgement and no temporal correlation of lossy links.

Beginning with the art gallery problem, there have been multiple efforts to determine an optimal configuration of sensors to cover a given region [41.71–73]. A variant which allows the use of mobile sensors is

known as the *watchmen tours* problem. In these approaches the sensor model is abstract and not well suited to real environments and cameras. Distributed geometric optimization methods [41.67] have also been used for mobile sensor network reconfiguration. A related class of methods is the use of estimation-theoretic optimization metrics and the application of information filters to coordinate network-wide motion [41.30]. There are other distributed optimization methods which use a distributed control law and show that it optimizes a global metric of interest, such as using a potential field or other linear control law based only on local neighbor interactions [41.74]. Research focusing on the control of cameras with pan, tilt, and zoom capabilities is due to [41.34, 75, 76]. In [41.75] an approach is developed to calibrate a pan-tilt-zoom camera automatically over its full zoom range and to build very high-resolution panoramas. In [41.34], the cameras are constantly moved to track observed targets, using a factor graph. A recent algorithm due to [41.77] significantly improves on this by positioning cameras to make the network better suited to detect and classify targets as they emerge. Pan-tilt-zoom cameras allow the construction of far more flexible vision systems than static cameras.

41.8 Control for Communication

In Sect. 41.5, we briefly discussed the benefits of using the communication network to synthesize and improve controller design. Conversely, the movement of robots affects the network and data transmission in the network. This gives rise to many challenges. If the controllers for individual robots are known, can we provide guarantees about communication in the network and can we develop robust information routing and networking algorithms in the presence of robot motion? Another challenge concerns how information propagates and diffuses in these networks. If the robots move under a given control model, how does a piece of information propagate through the network and what can we say about when and where that piece of information will be heard? If we know the answers to such questions, it may be possible to design controllers to realize desired communication network characteristics.

One simple control strategy that can affect network performance is to control the robot motion to ensure messages are transmitted between designated nodes. The movement of robots in a network of robots and sensors may cause network partitioning when nodes go out of

range. However, the ability of the robots to move in a controlled way also leads to an opportunity to address the information routing problem in disconnected networks by turning the robots into relay nodes. The key idea here is to enable the robot holding a current message to an unavailable destination to modify their trajectory in order to relay a message. This problem has been formulated as an optimization problem. The goal is to minimize the trajectory modifications necessary to send a message to its destination. Several solutions have been proposed depending on the information that is available to the robots. If the robots' trajectories are known, path planning techniques can be used to compute who moves where to relay what. If the robots' trajectories are not known, a distributed spanning tree can be created to enable robots to keep track of each other. Each robot is assigned a region of movement and a parent in the spanning tree. When the robot leaves its region, the parent is informed. When the robot moves too far away, the spanning tree is modified.

Mobile robots can be used to create desired network topologies under suitable models of network commu-

nication. If a robot is used to emplace nodes in an environment (or if sensor nodes robotically self-deploy) to build a network, the problem is referred to as deployment. It is possible to control the motion of individual nodes to guarantee that a specified topology is maintained [41.29]. It is also possible to repositioning nodes with the explicit aim of changing the network topology – the so-called mobility-based topology control problem.

A distributed algorithm for the deployment of mobile robot teams has been described by [41.78] using the concept of virtual pheromones: localized messages from one robot to another. These messages are used to generate either a gas expansion or a guided growth deployment model. Similar algorithms based on artificial potential fields are described in [41.79, 80], where the latter incorporates a connectivity constraint. An incremental deployment algorithm for mobile sensor networks is given in [41.32]; nodes are deployed one at a time into an unknown environment, with each node making use of information gathered by previously deployed nodes to determine its deployment location. The algorithm is designed to maximize network coverage while ensuring that nodes retain line of sight with one another.

Most work on network topology control has dealt with uncontrolled deployments, where there is no explicit control of the positions of individual nodes. The primary mechanisms proposed are power control and sleep scheduling. These methods involve pruning an already existing, well-connected communication graph in order to save power while ensuring that the resultant subgraph preserves connectivity. Given a network that is connected when all nodes are operating at maximum power, the aim of power control is to use the minimum power level at each node for which the network remains connected [41.81]. Given an overdeployed network, sleep scheduling seeks to activate a minimal subset of nodes to maintain connectivity and achieve other desired metrics [41.82]. In contrast, controlled deployments are feasible when the positions of individual nodes can be altered. Such deployments are interesting for two reasons. First, network topology with wireless communication relates directly to proximity relations and hence the position of the nodes. Second, there is increasing evidence that a large number of deployments are likely to involve careful, nonrandom placement of

nodes. The positioning of nodes is controlled either by the nodes themselves or by external agents. Such networks present a different and interesting scenario for topology control since it is possible to exploit control of the motion and placement of the nodes to build efficient topologies. A local, completely decentralized technique for topology control using mobility is given in [41.83].

An important application for networked robots is in monitoring and surveillance, where it is important that the robots cover the space while remaining within communication range [41.84]. Probing environment and adaptive sleeping protocol (PEAS) was one of the first attempts to address communication connectivity and sensing coverage simultaneously using heuristic algorithms [41.85]. Wang et al. [41.86] proposed a new coverage configuration protocol (CCP) to produce an approach that simultaneously optimizes coverage and connectivity while maximizing the number of nodes that are placed into sleep mode. Furthermore, they also identified three different classes of coverage–connectivity problems with respect to the ratio of radio and sensing ranges and recognized the critical ratio where the former range is twice as long as the latter. Zhang and Hou proved that, if the communication range is at least twice the sensing range, complete coverage of a convex area guarantees network communication connectivity, and then used this theorem as a basis for a localized density control algorithm [41.82]. This was subsequently generalized to show that the condition that the communication range is twice the sensing range is sufficient and is the tight lower bound to guarantee that complete coverage preservation implies communication connectivity among nodes if the original network topology is connected [41.87].

In summary, if the state of the communication network and the desired state of the communication network is known to each agent, it should be possible to synthesize distributed controllers to move agents to attain desired network characteristics. However, the assumptions on the global state are clearly not justified. Also, the desired motion to optimize network characteristics will conflict with the motion that is required to perform the desired task. However, as the brief discussion above illustrates, there are many interesting studies that point to promising directions for future work in this very fertile research field.

41.9 Conclusions and Further Reading

The paradigm of networked robots offers significant potential for accomplishing tasks that cannot be accomplished by individual robots. Indeed this paradigm is critical to such tasks as environmental monitoring, surveillance and reconnaissance, and security for civilian or defense purposes. However, there are many scientific challenges to realizing this vision for networked robots. The main overarching challenges are summarized here.

Technical challenges to scalability: We do not yet have a methodology for creating self-organizing robot networks that are robust to labeling (or numbering), with completely decentralized controllers and estimators, and with provable emergent response. This requires basic research at the intersection of control, perception, and communication.

Performing physical tasks in the real world: Most of our present applications emphasize going from static sensor networks to mobile sensor networks and, as such, are able to acquire and process information. We are a long way from creating robust robot networks that can perform physical tasks in the real world.

Human interaction for network-centric control and monitoring: Advances over the last decade have provided human users with the ability to interact with hundreds or thousands of computers on the Internet. It is necessary to develop similar network-centric approaches to interfacing, both for control and for monitoring.

Finally, a major challenge is to create robot networks that are proactive and anticipate our needs and commands rather than reacting (with delays) to human commands.

While there are major challenges ahead, there is no denying the tremendous potential of networked robots. This chapter and other related chapters in Part D (Chap. 31) and Part E (Chap. 40) show that the research community is making steady progress and confronting these challenges head on.

The references for this chapter includes many pointers to the specific research issues covered here. For some excellent further background on networked robotics we direct the reader to [41.5, 6, 19, 26, 29, 50, 64, 77].

References

- 41.1 Z. Butler, K. Kotay, D.L. Rus, M. Vona: Self-reconfiguring Robots, *Commun. ACM* **45**(3), 39–45 (2002)
- 41.2 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, *J. Robot. Syst.* **13**(11), 755–764 (1996)
- 41.3 L.E. Parker: The effect of heterogeneity in teams of 100+ mobile robots. In: *Multi-Robot Systems Volume II: From Swarms to Intelligent Automata*, ed. by A. Schultz, L.E. Parker, F. Schneider (Kluwer, Dordrecht 2003)
- 41.4 M. Yim, Y. Zhang, D. Duff: Modular robots, *IEEE Spect.* **39**(22), 30–34 (2002)
- 41.5 N. Hagita: Introduction to network robots, *Introduction to Network Robots* (Barcelona, 2005)
- 41.6 D. Estrin: *Embedded, Everywhere* (National Academies Press, Washington 2001)
- 41.7 J. Parrish, S. Viscido, D. Grunbaum: Self-organized fish schools: An examination of emergent properties, *Biol. Bull.* **202**, 296–305 (2002)
- 41.8 N. Franks, S. Pratt, E. Mallon, N. Britton, D. Sumpter: Information flow, opinion polling and collective intelligence in house-hunting social insects, *Philos. Trans. R. Soc. London. B* **357**, 1567–1584 (2002)
- 41.9 R.L. Jeanne: Group size, productivity, and information flow in social wasps. In: *Information Processing in Social Insects* (Birkhauser, Basel 1999)
- 41.10 Argo Floats: *A global array of 3,000 free-drifting profiling floats for environmental monitoring* (Argo Information Center, Ramonville 2007), <http://www.argo.ucsd.edu/index.html>
- 41.11 G.S. Sukhatme, A. Dhariwal, B. Zhang, C. Oberg, B. Stauffer, D.A. Caron: The design and development of a wireless robotic networked aquatic microbial observing system, *Environmen. Eng. Sci.* **24**(2), 205–215 (2006)
- 41.12 W. Kaiser, G. Pottie, M. Srivastava, G.S. Sukhatme, J. Villasenor, D. Estrin: Networked infomechanical systems (nims) for ambient intelligence. In: *Ambient Intelligence* (Springer, Berlin, Heidelberg 2005) pp. 83–114
- 41.13 Amarss: *Networked minirhizotron planning and initial deployment* (Center for Embedded Networkedsensing, Los Angeles 2007), <http://research.cens.ucla.edu/>
- 41.14 H. Singh, J. Catipovic, R. Eastwood, L. Freitag, H. Henriksen, F.F. Hover, D. Yoerger, J.G. Bellingham, B.A. Moran: An integrated approach to multiple AUV communications, navigation and docking, *IEEE Oceans* (1996) pp. 59–64

- 41.15 I. Vasilescu, M. Dunbabin, P. Corke, K. Kotay, D. Rus: Data collection, storage, and retrieval with an underwater sensor network, Proceedings of the ACM Sensys 2005 (San Diego, 2005)
- 41.16 N. Leonard, D. Paley, F. Lekien, R. Sepulchre, D.M. Fratantoni, R. Davis: Collective motion, sensor networks and ocean sampling, Proc. IEEE (2006)
- 41.17 D.O. Popa, A.S. Sanderson, R.J. Komerska, S.S. Mupparapu, D.R. Blidberg, S.G. Chappell: Adaptive sampling algorithms for multiple autonomous underwater vehicles, IEEE/OES AUV2004: A Workshop on Multiple Autonomous Underwater Vehicle Operations (2004)
- 41.18 M.A. Hsieh, A. Cowley, J.F. Keller, L. Chaimowicz, B. Grocholsky, V. Kumar, C.J. Taylor, Y. Endo, R. Arkin, B. Jung, D. Wolf, G. Sukhatme, D.C. MacKenzie: Adaptive teams of autonomous aerial and ground robots for situational awareness, J. Field Robot. **24**(11–12), 991–1014 (2007)
- 41.19 A. Howard, L.E. Parker, G.S. Sukhatme: Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection, Int. J. Robot. Res. **25**(5–6), 431–447 (2006)
- 41.20 L. Chaimowicz, A. Cowley, B. Grocholsky, M.A. Hsieh, J.F. Keller, V. Kumar, C.J. Taylor: Deploying air-ground multi-robot teams in urban environments, Third Multi-Robot Systems Workshop (Washington, 2005)
- 41.21 B. Grocholsky, R. Swaminathan, J. Keller, V. Kumar, G. Pappas: Information driven coordinated air-ground proactive sensing, IEEE International Conference on Robotics and Automation (IEEE, Barcelona 2005)
- 41.22 D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, B. Stewart: Distributed multi-robot exploration and mapping, Proc. IEEE (2006)
- 41.23 J. Seyfried, M. Szymanski, N. Bender, R. Estana, M. Theil, H. Worn: The I-Swarm Project: Intelligent Small World Autonomous Robots for Micro-manipulation, SAB 2004 International Workshop (Santa Monica 2004)
- 41.24 F. Mondada, G.C. Pettinaro, A. Guignard, I.W. Kwee, D. Floreano, J.-L. Deneubourg, S. Nofli, L.M. Gambardella, M. Dorigo: Swarm-Bot: A new distributed robotic concept, Auton. Robot. **17**, 193–221 (2004)
- 41.25 A. Ollero, S. Lacroix, L. Merino, J. Gancet, J. Wiklund, V. Remuss, I. Veiga, L.G. Gutierrez, D.X. Viegas, M.A. Gonzalez, A. Mallet, R. Alami, R. Chatila, G. Hommel, F.J. Colmenero, B. Arrue, J. Ferruz, R. Martinez de Dios, F. Caballero: Architecture and perception issues in the COMETS multi-UAV project, IEEE Robot. Autom. Mag. (2005)
- 41.26 Q. Li, D. Rus: Navigation protocols in sensor networks, ACM Trans. Sensor Netw. **1**(1), 3–35 (2005)
- 41.27 R. Bajcsy: Active perception, Proc. IEEE **76**, 996–1005 (1988)
- 41.28 T. Eren, D. Goldenberg, W. Whitley, Y.R. Yang, S. Morse, B.D.O. Anderson, P.N. Belhumeur: Rigidity, computation, and randomization of network localization, Proceedings of IEEE INFOCOM '04 (Hong Kong 2004)
- 41.29 A. Hsieh, A. Cowley, V. Kumar, C.J. Taylor: Towards the deployment of a mobile robot network with end-to-end performance guarantees, IEEE Int'l Conf. on Robotics and Automation (Orlando, 2006)
- 41.30 B. Grocholsky, S. Bayraktar, V. Kumar, C.J. Taylor, G. Pappas: Synergies in feature localization by air-ground teams, Proc. 9th Int'l Symposium of Experimental Robotics (Singapore 2004)
- 41.31 B. Grocholsky, E. Stump, V. Kumar: An extensive representation for range-only SLAM, International Symposium on Experimental Robotics (Rio de Janeiro, 2006)
- 41.32 A. Howard, M.J. Mataric, G.S. Sukhatme: An incremental self-deployment algorithm for mobile sensor networks, Auton. Robot. **13**(2), 113–126 (2002), Special Issue on Intelligent Embedded Systems
- 41.33 Z. Butler, D. Rus: Controlling mobile networks for monitoring events with coverage constraints, Proceedings of the IEEE International Conference on Robotics and Automation (2003)
- 41.34 M. Chu, J. Reich, F. Zhao: Distributed attention for large video sensor networks, Intelligent Distributed Surveillance Systems 2004 seminar (2004)
- 41.35 B. Jung, G.S. Sukhatme: Tracking targets using multiple robots: The effect of environment occlusion, Auton. Robot. **13**(3), 191–205 (2002)
- 41.36 R.M. Murray, K.J. Åström, S.P. Boyd, R.W. Brockett, G. Stein: Future directions in control in an information-rich world, IEEE Contr. Syst. Mag. (2003)
- 41.37 A. Pant, P. Seiler, K. Hedrick: Mesh stability of look-ahead interconnected systems, IEEE Trans. Autom. Control. **47**, 403–407 (2002)
- 41.38 T. Sugar, J. Desai, V. Kumar, J.P. Ostrowski: Coordination of multiple mobile manipulators, Proc. IEEE Int. Conf. Robot. Autom. **3**, 2022–2027 (2001)
- 41.39 R.W. Beard, J. Lawton, F.Y. Hadaegh: A coordination architecture for spacecraft formation control, IEEE Trans. Control Syst. Technol. **9**, 777–790 (2001)
- 41.40 A. Das, J. Spletzer, V. Kumar, C. Taylor: Ad hoc networks for localization and control, Proc. of the IEEE Conf. on Decision and Control (2002) pp. 2978–2983
- 41.41 J. Manyika, H. Durrant-Whyte: *Data Fusion and Sensor Management: An Information-Theoretic Approach* (Prentice Hall, Upper Saddle River 1994)
- 41.42 H.G. Tanner, A. Jadbabaie, G.J. Pappas: Stable flocking of mobile agents, part i: Fixed topology, Proceedings of the 42nd IEEE Conference on Decision and Control (2003) pp. 2010–2015
- 41.43 Q. Li, D. Rus: Navigation Protocols in Sensor Networks, ACM Trans. Sensor Netw. **1**(1), 3–35 (2005)

- 41.44 J.M. Fowler, R. D'Andrea: Distributed control of close formation flight, *Proc. of the IEEE Conf. on Decision and Control* (2002) pp.2972–2977
- 41.45 J.P. Desai, J.P. Ostrowski, V. Kumar: Modeling and control of formations of nonholonomic mobile robots, *IEEE Trans. Robot. Autom.* **17**(6), 905–908 (2001)
- 41.46 H.G. Tanner, V. Kumar, G.J. Pappas: Leader-to-formation stability, *IEEE Trans. Robot. Autom.* **20**(3) (2004)
- 41.47 S. Loizou, V. Kumar: Relaxed input to state stability properties for navigation function based systems, *Proceedings of the 45th IEEE Conference on Decision and Control* (San Diego, 2006)
- 41.48 M. Batalin, G.S. Sukhatme: Coverage, exploration and deployment by a mobile robot and communication network, *Telecommun. Syst. J.* **26**(2), 181–196 (2004), Special Issue on Wireless Sensor Networks
- 41.49 K. J. O'hara, V. Bigio, S. Whitt, D. Walker, T. R. Balch: *Evaluation of a Large Scale Pervasive Embedded Network for Robot Path Planning* (ICRA 2006) pp. 2072–2077
- 41.50 V. Kumar, N. Leonard, A.S. Morse (eds.): *Cooperative Control, Vol. 309 of Lecture Notes in Control and Information Sciences* (Springer, Berlin, Heidelberg 2004)
- 41.51 J. Chen, S. Teller, H. Balakrishnan: Pervasive pose-aware applications and infrastructure, *IEEE Comput. Graph. Appl.* **23**(4), 14–18 (2003)
- 41.52 A. Savvides, C.-C. Han, M. Srivastava: Dynamic Fine-Grained localization in Ad-Hoc networks of sensors, *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking (MOBICOM-01)* (ACM, New York 2001) pp.166–179
- 41.53 D. Moore, J. Leonard, D. Rus, S.J. Teller: Robust distributed network localization with noisy range measurements, *SenSys '04: Proceedings of the 2nd International conference on Embedded networked sensor systems* (2004) pp. 50–61
- 41.54 C. Detweiler, J. Leonard, D. Rus, S. Teller: Passive Mobile Robot Localization within a Fixed Beacon Field, *Proceedings of the 2006 International Workshop on Algorithmic Foundations of Robotics* (2006)
- 41.55 N. Bulusu, J. Heidemann, D. Estrin: Adaptive beacon placement, *Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS-01)* (Los Alamitos 2001) pp. 489–498
- 41.56 S.N. Simic, S. Sastry: *Distributed localization in wireless ad hoc networks*, Technical Report UCB/ERL M02/26, EECS Department (University of California, Berkeley 2001), <http://www.eecs.berkeley.edu/Pubs/TechRpts/2002/4010.html>
- 41.57 P. Corke, R. Peterson, D. Rus: Communication-assisted localization and navigation for networked robots, *Int. J. Robot. Res.* **4**(9), 116 (2005)
- 41.58 R. Nagpal, H.E. Shrobe, J. Bachrach: Organizing a global coordinate system from local information on an ad hoc sensor network, *Lect. Notes Comput. Sci.* **2634**, 333–348 (2003)
- 41.59 A. Howard, M.J. Mataric, G.S. Sukhatme: Putting the 'i' in 'team': An ego-centric approach to cooperative localization, *IEEE International Conference on Robotics and Automation* (Taipei, 2003) pp. 868–892
- 41.60 A. Howard, M.J. Mataric, G.S. Sukhatme: Localization for mobile robot teams using maximum likelihood estimation, *IEEE/RSJ International Conference on Intelligent Robots and Systems* (Lausanne, 2002) pp. 434–459
- 41.61 R. Vidal, O. Shakernia, H.J. Kim, D.H. Shim, S. Sastry: Probabilistic pursuit-evasion games: Theory, implementation and experimental evaluation, *IEEE Trans. Robot. Autom.* **18**(5), 662–669 (2002)
- 41.62 M. Batalin, M.H. Rahimi, Y. Yu, D. Liu, A. Kansal, G.S. Sukhatme, W. Kaiser, M. Hansen, G. Pottie, M. Srivastava, D. Estrin: Call and response: Experiments in sampling the environment, *ACM SenSys* (Baltimore, 2004) pp. 25–38
- 41.63 M.H. Rahimi, W. Kaiser, G.S. Sukhatme, D. Estrin: Adaptive sampling for environmental field estimation using robotic sensors, *IEEE/RSJ International Conference on Intelligent Robots and Systems* (2005) pp. 747–753
- 41.64 F. Zhao, L. Guibas: *Wireless Sensor Networks: An Information Processing Approach* (Morgan Kaufmann, New York 2004)
- 41.65 F. Zhang, B. Grocholsky, V. Kumar: Formations for localization of robot networks, *IEEE Int'l Conf. on Robotics and Automation* (New Orleans, 2004)
- 41.66 J. Cortes, S. Martinez, T. Karatas, F. Bullo: Coverage control for mobile sensing networks, *IEEE Trans. Robot. Autom.* **20**(2), 243–255 (2004)
- 41.67 J. Cortes, S. Martinez, F. Bullo: Spatially-distributed coverage optimization and control with limited-range interactions, *ESAIM Contr. Optim. Calc. Variat.* **11**, 691–719 (2005)
- 41.68 M. Schwager, J. McLurkin, D. Rus: Distributed coverage control with sensory feedback for networked robots, *Proc. Robot. Sci. Syst. (RSS)* (2006)
- 41.69 M. Schwager, J.-J. Slotine, D. Rus: Decentralized Adaptive Control for Coverage for Networked Robots, *Proceedings of the 2007 International Conference on Robotics and Automation* (2007)
- 41.70 A. Krause, C. Guestrin, A. Gupta, J. Kleinberg: Near-optimal sensor placements: Maximizing information while minimizing communication cost, *Fifth International Conference on Information Processing in Sensor Networks (IPSN'06)* (2006)
- 41.71 V. Chvatal: A combinatorial theorem in plane geometry, *J. Combin. Theory Ser.* **18**, 39–41 (1975)
- 41.72 J. O'Rourke: *Art Gallery Theorems and Algorithms* (Oxford Univ. Press, New York 1987)

- 41.73 S. Fisk: A short proof of Chvatal's watchmen theorem, *J. Combin. Theory Ser. 24*, 374 (1978)
- 41.74 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, *IEEE Trans. Autom. Control* **48**(6), 988–1001 (2003)
- 41.75 S. Sinha, M. Pollefeys: Camera network calibration from dynamic silhouettes, *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition* (2004)
- 41.76 R. Collins, A. Lipton, H. Fujiyoshi, T. Kanade: Algorithms for cooperative multisensor surveillance, *Proc. IEEE* (2001)
- 41.77 A. Kansal, W. Kaiser, G. Pottie, M. Srivastava, G.S. Sukhatme: Reconfiguration methods for mobile sensor networks. Submitted to *ACM Transactions on Sensor Networks* (2006)
- 41.78 D. Payton, M. Daily, R. Estkowski, M. Howard, C. Lee: Pheromone robotics, *Auton. Robot.* **11**, 319–324 (2001)
- 41.79 A. Howard, M.J. Matarić, G.S. Sukhatme: Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem, *Proceedings of the International Symposium on Distributed Autonomous Robotic Systems* (2002) pp. 299–308
- 41.80 S. Poduri, G.S. Sukhatme: Constrained coverage for mobile sensor networks, *IEEE International Conference on Robotics and Automation* (New Orleans 2004) pp.165–172
- 41.81 R. Wattenhofer, L. Li, P. Bahl, Y.M. Wang: A cone-based distributed topology-control algorithm for wireless multi-hop networks, *IEEE/ACM Trans. Netw.* (2005)
- 41.82 H. Zhang, J.C. Hou: On deriving the upper bound of alpha-lifetime for large sensor networks, *ACM Trans. Sensor Netw.* **1**(6) (2005)
- 41.83 S. Poduri, S. Patten, B. Krishnamachari, G.S. Sukhatme: A unifying framework for tunable topology control in sensor networks. Technical Report CRES-05-004 (2005)
- 41.84 M.A. Hsieh, V. Kumar: Pattern generation with multiple robots, *IEEE Int'l Conf. on Robotics and Automation* (Orlando, 2006)
- 41.85 F. Ye, G. Zhong, J. Cheng, L. Zhang, S. Lu: Peas: A robust energy conserving protocol for long-lived sensor networks, *International Conference on Distributed Computing Systems* (2003)
- 41.86 X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, C. Gill: Integrated coverage and connectivity configuration in wireless sensor networks, *First ACM Conf. on Embedded Networked Sensor Systems, SenSys* (2003)
- 41.87 D. Tian, N.D. Georganas: Connectivity maintenance and coverage preservation in wireless sensor networks. *AdHoc Networks* (2005)

Field and Part F

Part F Field and Service Robotics

42 Industrial Robotics

Martin Hägele, Stuttgart, Germany
Klas Nilsson, Lund, Sweden
J. Norberto Pires, Coimbra, Portugal

Ed. by Alexander Zelinsky

43 Underwater Robotics

Gianluca Antonelli, Cassino, Italy
Thor I. Fossen, Trondheim, Norway
Dana R. Yoerger, Woods Hole, USA

44 Aerial Robotics

Eric Feron, Atlanta, USA
Eric N. Johnson, Atlanta, USA

45 Space Robots and Systems

Kazuya Yoshida, Sendai, Japan
Brian Wilcox, Pasadena, USA

46 Robotics in Agriculture and Forestry

John Billingsley, Toowoomba, Australia
Arto Visala, Helsinki, Finland
Mark Dunn, Toowoomba, Australia

47 Robotics in Construction

Kamel S. Saidi, Gaithersburg, USA
Jonathan B. O'Brien, Sydney, Australia
Alan M. Lytle, Gaithersburg, USA

48 Robotics in Hazardous Applications

James P. Trevelyan, Perth, Australia
Sung-Chul Kang, Seoul, Korea
William R. Hamel, Knoxville, USA

49 Mining Robotics

Peter Corke, Kenmore, Australia
Jonathan Roberts, Kenmore, Australia
Jock Cunningham, Kenmore, Australia
David Hainsworth, Kenmore, Australia

50 Search and Rescue Robotics

Robin R. Murphy, Tampa, USA
Satoshi Tadokoro, Sendai, Japan
Daniele Nardi, Roma, Italy
Adam Jacoff, Gaithersburg, USA
Paolo Fiorini, Verona, Italy
Howie Choset, Pittsburgh, USA
Aydan M. Erkmen, Ankara, Turkey

51 Intelligent Vehicles

Alberto Broggi, Parma, Italy
Alexander Zelinsky, Epping, Sydney, Australia
Michel Parent, Le Chesnay, France
Charles E. Thorpe, Pittsburgh, USA

52 Medical Robotics and Computer-Integrated Surgery

Russell H. Taylor, Baltimore, USA
Arianna Menciassi, Pisa, Italy
Gabor Fichtinger, Kingston, Canada
Paolo Dario, Pisa, Italy

53 Rehabilitation and Health Care Robotics

H.F. Machiel Van der Loos, Vancouver, Canada
David J. Reinkensmeyer, Irvine, USA

54 Domestic Robotics

Erwin Prassler, Sankt Augustin, Germany
Kazuhiro Kosuge, Sendai, Japan

55 Robots for Education

David P. Miller, Norman, USA
Illah R. Nourbakhsh, Pittsburgh, USA
Roland Siegwart, Zürich, Switzerland

Part F, Field and Service Robotics, covers the most recent advances concerned with the growing area of robot applications, ranging from industrial robotics, through a diverse array of air, land, sea, and space applications to educational robotics. The future vision for robotics is for the pervasive application of robots. The robots of the future will perform all the dangerous, dirty, and dreary (DDD) tasks. Joe Engelberger, the pioneer of the robotics industry, wrote in his 1989 book *Robotics in Service* that the inspiration to write the book came as a reaction to a forecast study of robot applications, which predicted that in 1995 applications of robotics outside factories (the traditional domain of industrial robots) would account for less than 1% of total sales. Engelberger believed that this forecast was wrong, and he instead predicted that the non-industrial class of robot applications would become the largest class of robot applications. Engelberger's prediction has yet to come to pass. However, he did correctly foresee the growth in nontraditional applications of robots. Previous parts of this Handbook show the great strides that robotics technology has made in the past 50 years. The technology has reached a level of maturity such that robots are now marching from the factories into field and service applications.

The topics in Part F cover the essentials of what is required to create robots that can work in all environments. Part F describes fit-for-purpose robots and includes hardware design, control (of locomotion, manipulation, and interaction), perception, and user interfaces. The economic/social drivers for the particular applications are also discussed. Part F builds on all of the previous parts of the Handbook. Robot Foundations (Part A) and Robot Structures (Part B) are essential for providing the basic mechanisms and control structures for any robot that is targeted for applications work. Sensing and Perception (Part C) and Manipulation and Interfaces (Part D) are critical capabilities for robots that need to interact with changing environments and perform manipulation tasks under human supervision. To fully utilize the potential of a working robot requires mobility and the ability to cooperate with other robotics, the basic technologies described in Mobile and Distributed Robotics (Part E) are essential.

Chapter 42, Industrial Robotics, gives a history and description of typical industrial robot applications. Most robots today can trace their origin to early industrial robot designs. All the important foundations for robot control were initially developed with industrial applications in mind. The chapter describes how robots

with different mechanisms are designed to fit different applications.

Chapter 43, Underwater Robotics, deals with challenges of technology to support marine applications. A historical background showing the expanding role of robotics in oceanic engineering applications is given. The chapter deals with modeling underwater dynamic effects, sensors and actuators for marine robots, mission control systems, guidance and control algorithms. Underwater localization and manipulation are discussed. Fault detection and fault tolerance aspects, together with the coordination control of multiple underwater vehicles are also covered. The chapter reports on successful deployments of marine robotics.

Chapter 44, Aerial Robotics, describes the technology, challenges, and applications of airborne robotics. The wide range of potential uses for robots that can fly, from military to civilian applications, is described. The challenges for aerial robots span many fields, including state regulations, man-machine interface design issues, path planning, navigation, safety/reliability, collision prevention, take-off/landing techniques, and dealing with other aerial vehicles. Basic aerial robot flight concepts including propulsion systems, control systems, and flight vehicle types including lighter-than-air systems are covered. The chapter ends with a description of integrated automated aerial robotics systems.

Chapter 45, Space Robots and Systems, deals with robotics technology for space applications. It covers two classes of robotics: orbital and surface. Any unmanned spacecraft is a robotic spacecraft. However, space robots are more capable devices that can facilitate manipulation, assembly, assisting astronauts, or extending exploration to remote planets as surrogates for human explorers. This chapter covers the key issues in space robots and systems: manipulation, mobility, sensing and perception, teleoperation and autonomy, and dealing with extreme environments. A historical overview and a review of technical advances for space robots is provided. A description of the mathematics for the control of robotic devices in microgravity environments is also given.

Chapter 46, Robotics in Agriculture and Forestry, describes how robotics has made a impact in the agricultural and forestry industries. Automatic vehicle guidance of farming machinery are being widely used. Automatic sensing, handling, and processing of produce are commonplace, and there has been a substantial automation of processes and procedures for livestock handling. In forestry, legged harvesters and automation of trimming are maturing technologies. Computer vision is being

applied to allow tractors to follow a row of crops, collecting macadamia nuts, and for sorting and grading of produce. A number of farm automation projects, such as automated milking of cows or sorting sheep from goats using a swinging gate to separate different species using computer vision are a commercial success. The chapter emphasizes the agricultural field as a fruitful source for robotics applications, some sufficiently demanding to require the development of new theoretical techniques and methods.

Chapter 47, Robotics in Construction, describes the construction automation concepts that have been developed and presents examples of construction robots that are in use and/or in various stages of development. The chapter gives an overview of the construction industry and discusses the concept of automation versus robotics. Sensing systems that enable the use of robots and automation in construction are reviewed. The chapter also discusses economic aspects of implementing robotics in construction. Unsolved technical challenges including interoperability, connection systems, tolerances, power, and communications are discussed.

Chapter 48, Robotics in Hazardous Applications, discusses robot technologies for dealing with difficult and dangerous environments. The technology solutions that are adopted depend on the nature and magnitude of the hazards. When the magnitudes of hazards reach the point that human exposure represents a direct threat to life or has long-term health consequences, e.g., nuclear radiation, some form of remote operation must be used. This chapter describes robotics technologies for dealing with eradicating landmines, and hazardous materials handling, e.g., explosives disposal, and handling of dangerous biological and nuclear material. Enabling technologies to support demining and hazardous material handling are discussed, including mobile platforms, manipulator design, teleoperation and control, and the supply of energy and communication signals through reliable and robust tethers. The chapter ends with a discussion of the advances that are required to progress the field. Progress is needed in mechatronics design, sensing, machine intelligence, and fully understanding the application and its associated economic factors.

Chapter 49, Mining Robotics, discusses the application of robotics to the mining industry. Mining remains physically demanding and hazardous work. The scope for the application of robotics is great since mining requires the handling of enormous quantities of material in a cost-effective and safe manner. High operational costs, the need for greater productivity, and improved health and safety outcomes are powerful drivers for

robotics. Existing approaches to improving productivity and safety through training, work practices, and larger and improved machine design are providing diminishing returns. The industry is at a tipping point where robotics and automation will provide the next step change in productivity and safety. This chapter reviews the history of robotic mining; it describes the enabling robot technologies associated with metalliferous mining and coal mining (open-cut and underground).

Chapter 50, Search and Rescue Robotics, describes the state of the art of an emerging applications field. Rescue robots serve as extensions of responders to a disaster, providing real-time video and other sensory data about the situation. The technology is still emerging and is still to be adopted by the international emergency response community. Rescue robots are beginning to be used by fire departments and for bomb disposal. This chapter describes the basic characteristics of disasters and the impact on robotic design, describes the robots actually used in disasters to date, promising robot designs, and concepts and methods of evaluation in benchmarks for rescue robotics. The chapter discusses the fundamental challenges facing rescue robotics, including the key challenge of evolution from a concept to the adoption as a solution technology.

Chapter 51, Intelligent Vehicles, describes the emerging robotics application field of intelligent vehicles: motor vehicles that have autonomous functions and capabilities. The chapter describes why the development of intelligent vehicles is important and gives a brief history of the field and the potential benefits of the technology. It describes the enabling technologies for intelligent vehicles to sense vehicle, environment, and driver state, work with digital maps and satellite navigation, and communicating with intelligent transportation infrastructure. The chapter describes the challenges and solutions associated with road scene understanding. Advanced driver assistance systems which use robotics technologies to create new safety and convenience systems for motor vehicles, such as collision avoidance, lane keeping, parking assistance, as well as driver monitoring to mitigate against driver fatigue, inattention, and impairment are described.

Chapter 52, Medical Robots and Systems, describes the development of medical robotics. In the last 20 years starting with brain surgery, orthopaedics, endoscopic surgery, and microsurgery, the field has expanded to include commercially marketed, clinically deployed systems and an active research community. This chapter provides a historical review of the field and discusses the major thrusts using examples from current and past

research. Medical robots are described within the context of larger computer-integrated systems including presurgical planning, intraoperative execution, and post-operative assessment and follow-up. The basic concepts of computer-integrated surgery including the critical factors affecting the deployment and acceptance of medical robots are described. An overview of medical robot systems is provided, including remote telesurgery and robotic surgical simulators.

Chapter 53, Rehabilitation and Health Care Robotics, describes robotic systems that assist persons who have a disability or provide rehabilitation therapy for persons. The chapter provides a historical review of the field. Physical therapy and training robots as well as robotic aids for people with disabilities are described. Advances in smart prostheses and orthoses for rehabilitation robotics are also described. An overview of work in diagnosis and monitoring for rehabilitation is provided. The chapter provides a solid understanding of the future challenges for rehabilitation and health care robotics.

Chapter 54, Domestic Robots, describes the technology that everyone would one day like in their home, that will vacuum the house, clean the kitchen, load the dishwasher, or polish the shoes. In spite of hundreds of millions of potential users surprisingly not many domestic robots exist. This chapter examines what it means not only to develop but also to commercialize domestic robots using cleaning robots as a representative example. The chapter reviews domestic floor cleaning

robots, robotic pool cleaners, window cleaning robots, and robotic lawn mowing. The chapter also examines smart appliances such as ironing robotics, intelligent refrigerators, and digital wardrobes. A description of research projects on smart environments and smart homes is presented. The chapter concludes on the importance of the economic case for the development of service robots.

Chapter 55, Robots for Education, describes robotics technologies for the education market, both as motivational tools for research programs and as concrete, real-world deployments that demonstrate the state of the art in robotic technologies. This chapter provides a technology overview of education robots. The chapter also provides a survey of robot-themed tournaments and an overview of robotic installations in informal learning spaces. Robotic technology is being used in hands-on, active-learning museums; this requires a new level of technology robustness and standardization. The chapter discusses physical robot platforms that have successfully supported active learning. An important tool in the study and execution of educational robotic systems is the formal evaluation of the efficacy of a robot system in an educational context. The chapter summarizes the manner in which conventional analytical tools may be used to evaluate educational programs that use robotic technologies as learning tools.

Sincere thanks and acknowledgement for help and assistance in preparing Part F are extended to Vivian Cateaux and Philippa V'Landys.

Industrial Robotics

Martin Hägele, Klas Nilsson, J. Norberto Pires

Most robots today can trace their origin to early industrial robot designs. Much of the technology that makes robots more human-friendly and adaptable for different applications has emerged from manufacturers of industrial robots. Industrial robots are by far the largest commercial application of robotics technology today. All the important foundations for robot control were initially developed with industrial applications in mind. These applications deserve special attention in order to understand the origin of robotics science and to appreciate many unsolved problems that still prevent the wider use of robots in manufacturing. In this chapter we present a brief history and descriptions of typical industrial robotics applications. We show how robots with different mechanisms fit different applications. Even though robots are well established in large-scale manufacturing, particularly in automobile and related component assembly, there are still many challenging problems to solve. The range of feasible applications could significantly increase if robots were easier to install, to integrate with other manufacturing processes,

and to program, particularly with adaptive sensing and automatic error recovery. We outline some of these remaining challenges for researchers.

Industrial robots are considered as a cornerstone of competitive manufacturing, which aims to combine high productivity, quality, and adaptability at minimal cost. In 2007 more than one million industrial robot installations were reported, with automotive industries as the predominant users with a share of more than 60% [42.1]. However, high-growth industries (in life sciences, electronics, solar cells, food, and logistics) and emerging manufacturing processes (gluing, coating, laser-based processes, precision assembly etc.) will increasingly depend on advanced robot technology. These industries' share of the number of robot installations has been growing steadily.

42.1 A Short History of Industrial Robots	964
42.2 Typical Applications and Robot Configurations	969
42.2.1 Welding	969
42.2.2 Car Body Assembly	969
42.2.3 Painting.....	971
42.2.4 Material Transfer Automation	971
42.2.5 Machining.....	974
42.2.6 Human-Robot Cooperation for Handling Tasks.....	974
42.3 Kinematics and Mechanisms	975
42.4 Task Descriptions – Teaching and Programming	976
42.5 End-Effectors and System Integration	980
42.6 Conclusions and Long-Term Challenges ..	983
References	985

The production of industrial robots on the one hand, and the planning, integration, and operation of robot workcells on the other hand, are largely independent engineering tasks. In order to be produced in sufficiently large quantities, a robot design should meet the requirements for the widest set of potential applications. As this is difficult to achieve in practice, various classes of robot designs regarding payload capacity, number of robot axes, and workspace volume have emerged for application categories such as assembly, palletizing, painting, welding, machining, and general handling tasks.

Generally, a robot workcell consists of one or more robots with controllers and robot peripherals: grippers

or tools, safety devices, sensors, and material transfer components for moving and presenting parts. Typically, the cost of a complete robot workcell is four times the cost of the robots alone.

A robot workcell is usually the result of customized planning, integration, programming, and configuration, requiring significant engineering expertise. Standardized engineering methods, tools, and best-practice examples have become available to reduce costs and provide more predictable performance [42.2].

Today's industrial robots are mainly the result of the requirements of capital-intensive large-volume manufacturing, mainly defined by the automotive, electronics, and electrical goods industries. Future industrial robots will not be a mere extrapolation of today's designs with respect to features and performance data, but will rather follow new design principles addressing a wider range of application areas and industries. At the same time, new technologies, particularly from the information technol-

ogy (IT) world, will have an increasing impact on the design, performance, and cost of future industrial robots.

International and national standards now help to quantify robot performance and define safety precautions, geometry, and media interfaces. Most robots operate behind secure barriers to keep people at a safe distance [42.3]. Recently, improved safety standards have allowed direct human–robot collaboration, permitting robots and human factory workers to share the same workspace [42.4].

We will first present a historical introduction to industrial robotics with a selection of contemporary application examples, then the basic principles that are used in industrial robotics and a review of programming methods will be presented. We will also discuss tools (end-effectors) and system integration requirements. The chapter will be closed with the presentation of selected, unsolved problems that currently inhibit the wider application of industrial robots.

42.1 A Short History of Industrial Robots

The invention of the industrial robot dates back to 1954 when George Devol filed a patent on a *programmed article transfer* (Fig. 42.1). After teaming up with Joseph Engelberger, the first robot company, Unimation, was founded and put the first robot into service at a General Motors plant in 1961 for extracting parts

from a die-casting machine. Most of the hydraulically actuated *Unimates* were sold through the following years for workpiece handling and for spot-welding of car bodies [42.5]. Both applications were successful, which means that the robots worked reliably and ensured uniform quality. Soon, many other companies

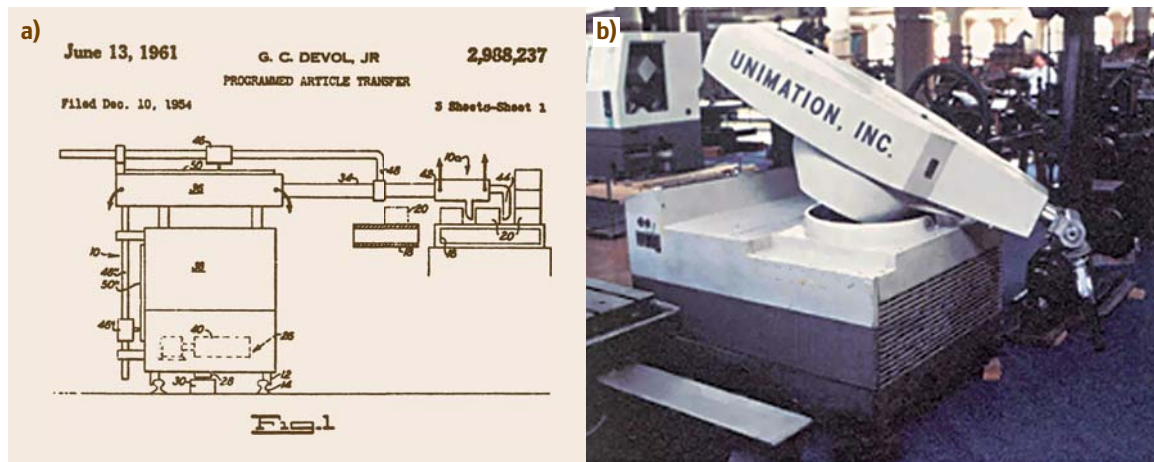


Fig. 42.1a,b The invention of the industrial robot (a) This patent was the start of a joint effort of G. Devol and J. Engelberger to form the first robot company, Unimation, a fusion of the words universal and automation. The company was acquired by Westinghouse in the late 1980s. (b) The first Unimation performed a rather simple handling task in 1961 at a General Motors plant. Other car manufacturers followed. The photo shows the first robot installed at Ford from their Museum in Dearborn

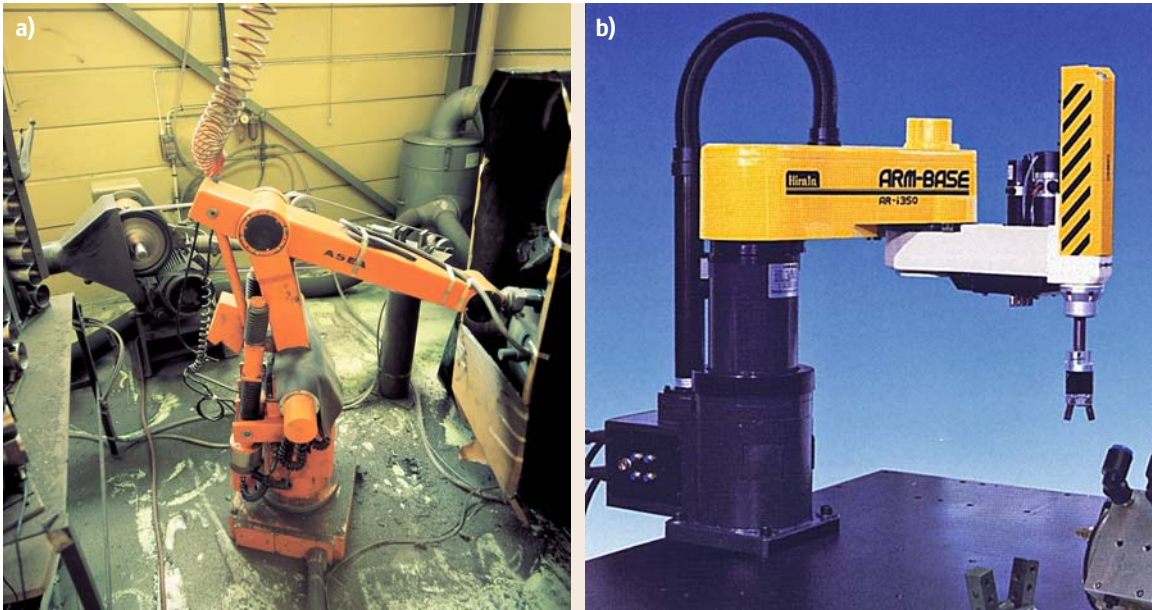


Fig. 42.2a,b The all electric (a) IRB-6 and (b) a SCARA-type kinematic (a) First introduced in 1973, the IRB-6 has been a breakthrough development as it was the first serially-produced robot product, which combined all-electric-drives technology and a microcomputer for programming and motion control. The robot proved very robust. Life-times of more than 20 years in harsh productions were reported (by courtesy ABB Automation, Friedberg) (b) The *selective compliance assembly robot arm* (SCARA) is particularly suited for assembly tasks as it combines rigidity in the vertical axis and compliance in the horizontal axis. In 1978, the first Hirata AR-300 was put together. Depicted is the successor design, the AR-i350. The SCARA design combines three or four rotational and one translational axis (by courtesy HIRATA Robotics, Mainz)

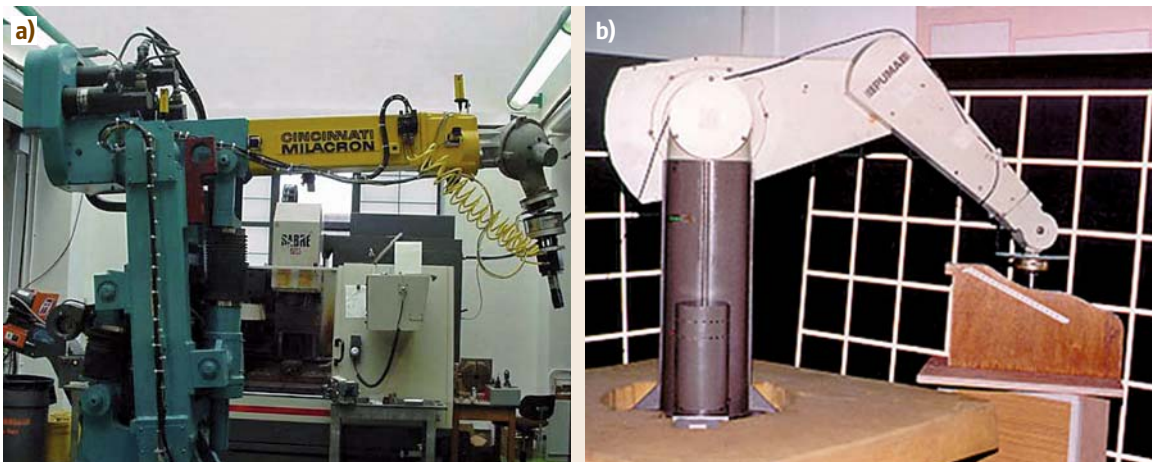


Fig. 42.3a,b Cincinnati Milacron T3 and the Unimation PUMA 560 (a) In 1974, Cincinnati Milacron introduced the first microcomputer controlled robot. The first T3 (*The Tomorrow Tool*) models used hydraulic drives, later they were replaced by electric motors as shown in the photo. The CM robotics division was acquired by ABB in the late 1970s (b) This 6 axis PUMA (programmable universal machine for assembly) came close to the dexterity of a human arm. After its launch in 1979 by Unimation it became one of the most popular arms and was used as a reference in robotics research for many years

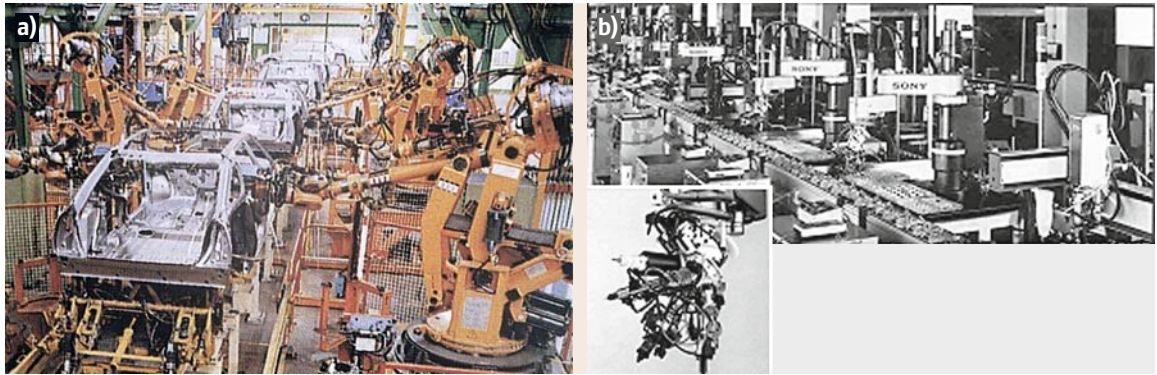


Fig. 42.4a,b Spot-welding line and videocassette recorder (VCR) assembly as pioneering applications for industrial robot applications. (a) Spot welding quickly became a primary application for robots as these jobs were particularly exhausting and hazardous for workers. A typical car body welding line from 1985 is displayed. The car model shown is a French Citroën CX (b) An automated VCR assembly line (ca. 1989) with SCARAs carrying a turret with multi-gripper tools. Typically five parts are added by one robot before the VCR is moved to the next station of the fully automated assembly line

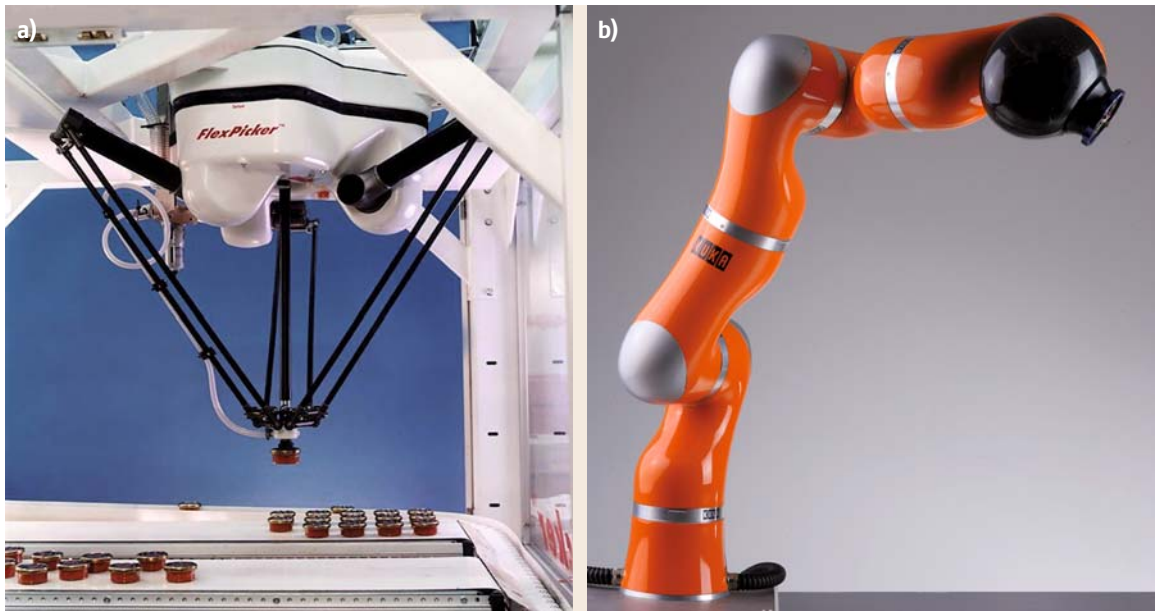


Fig. 42.5a,b The ABB FlexPicker parallel robot for high speed picking and the KUKA light weight robot arm (a) *Parallel kinematic machines (PKM)* represent an interesting approach to achieve high stiffness at low inertia, thus allowing accurate, high speed motions. Initially suggested by Clavel this 4 axis robot (called Delta) is used for high speed pick-and-place tasks. The robot reaches accelerations of up to 10 g (b) The KUKA light-weight robot is the result of a long research and development process by DLR, towards an arm design with a weight-to-payload ratio of 1:1. The 7 axis arm which is suited for human-robot cooperation imitates the dexterity of a human arm

started to develop and manufacture industrial robots. An innovation-driven industry was born. However, it took many years until this industry became profitable.

The breakthrough Stanford Arm was designed as a research prototype in 1969 by Victor Scheinman (see Chap. 3), at that time a mechanical engineering stu-

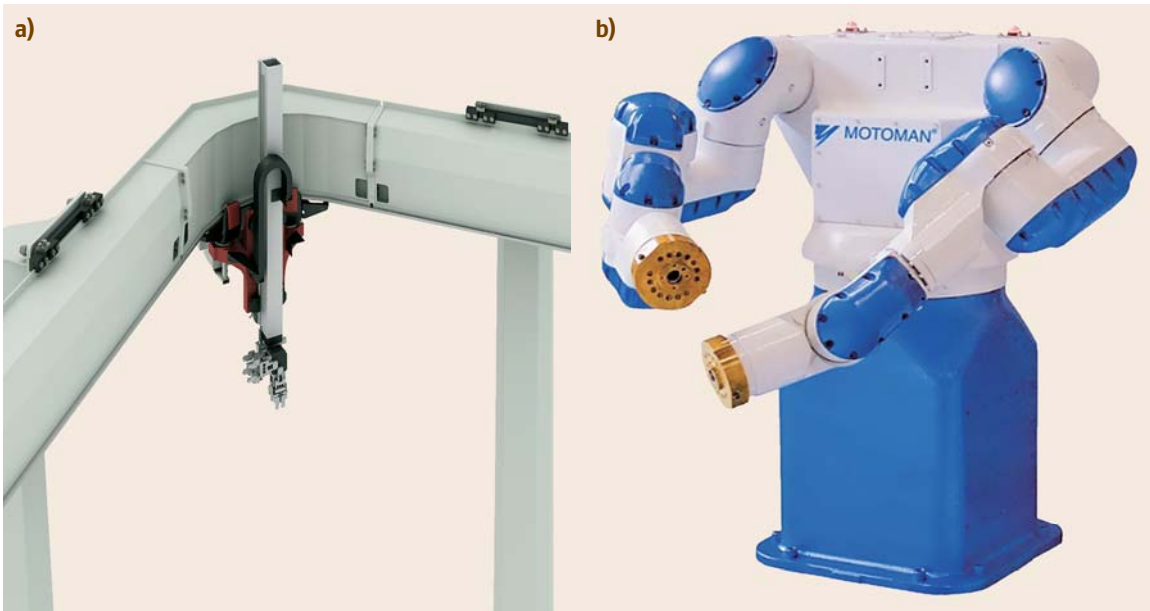


Fig. 42.6a,b Cartesian robot mobility and two-armed robot dexterity (a) The roboLoop of Güdel is a curved-track gantry and transfer system. One or more robot arms circulate as carriers in a closed transfer system. The system can be installed suspended, in gantry configuration, or as a floor-standing system. A signal bus allows the control and coordination of multiple robo-carriers (by courtesy Güdel, Langenthal) (b) Motoman's DA-20 dual-arm robot provides high-speed motion with two six-axis arms that provide enhanced, human-like flexibility of movement. The robot also provides jig-less operation with one robot arm holding a part while the other performs operations on the held part (by courtesy Motoman, Kalmar)

dent working in the Stanford Artificial Intelligence Laboratory (SAIL) [42.6]. The six-degree-of-freedom (6-DOF) all-electric manipulator was controlled by a standard computer, a Digital Equipment PDP-6. The non-anthropomorphic kinematic configuration with one prismatic and five rotational joints was configured such that the equations for solving the robot kinematics were simple enough to speed up computations. Drives consisted of direct-current (DC) electric motors, harmonic drive and spur gear reducers, potentiometers and tachometers for position and velocity feedback. Subsequent robot designs were strongly influenced by Scheinman's concepts (Figs. 42.2 and 42.3).

In 1973, the company ASEA (now ABB) introduced the first microcomputer-controlled all-electric industrial robot, the IRB-6, which allowed continuous path motion, a precondition for arc-welding or machining (Fig. 42.2). The design proved to be very robust and robot lifetimes of more than 20 years were reported [42.7]. In the 1970s intense diffusion of robots into car manufacturing set in mostly for (spot-)welding and handling applications (Fig. 42.4).

In 1978, the *selective compliance assembly robot arm* (SCARA) was invented by Hiroshi Makino of Yamaguchi University, Japan [42.8]. The ground-breaking four-axis low-cost design was perfectly suited for small parts assembly as the kinematic configuration allows fast and compliant arm motions (Fig. 42.2). Flexible assembly systems based on the SCARA robot in conjunction with compatible product designs (design for assembly, DFA) have contributed significantly to creating a worldwide boom in high-volume electronics production and consumer products [42.9].

Requirements regarding a robot's speed and weight have led to novel kinematic and transmission designs. From early on, the reduction of the mass and inertia of robot structures was a primary research target, where the human arm with a weight-to-load ratio of 1:1 was considered the ultimate benchmark. In 2006, this goal was reached in the form of the company KUKA lightweight robot, a compact 7-DOF robot arm with advanced force-control capabilities [42.10]. Another approach towards lightweight and stiff structures has been pursued since the 1980s by developing parallel kinematic machines

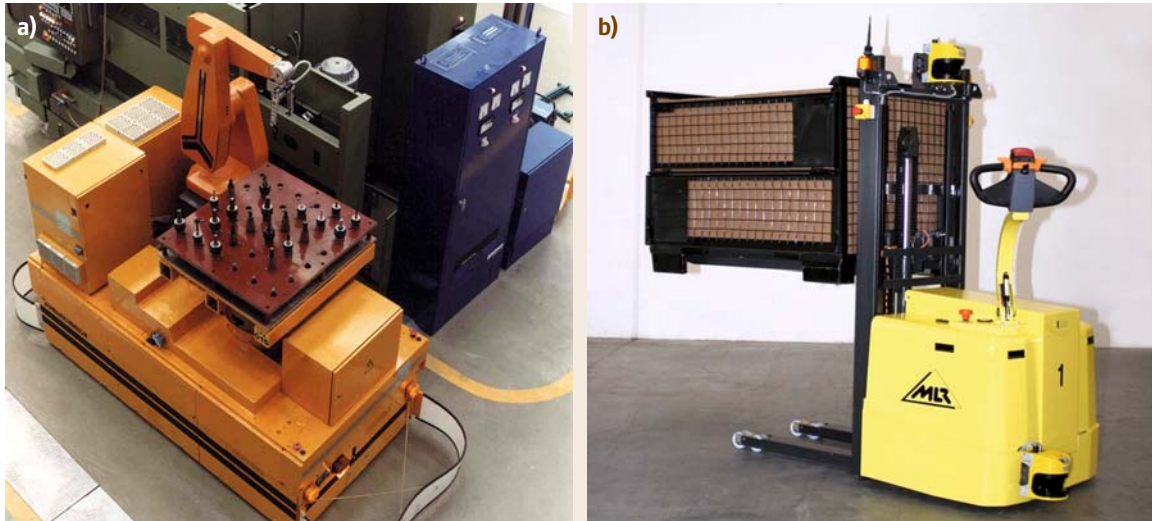


Fig. 4.2.7a,b A mobile robot arm for machine (un)loading and an autonomous fork lift (a) The MORO (mobile robot, 1984), developed at Fraunhofer IPA, Stuttgart, was one of the first prototypes to combine a robot arm on a wire-bound mobile platform which follows a wire buried in the floor. Mobile arms were particularly used in highly automated clean room manufacturing facilities to automatically load and unload process equipment (b) This automated fork-lift by MLR combines manual operation with fully autonomous navigation and (un-)loading of boxes and pallets. A laser scanner using retro-reflecting tapes or other landmarks (walls, pillars) avoids constraints imposed by simple buried wire guidance technology (by courtesy MLR System, Ludwigsburg)

which connect the machine's basis with its end-effector by three to six parallel struts [42.11]. These so-called parallel robots (see Chaps. 3 and 12) are particularly suited to achieve the highest speeds (e.g., for picking), precision (e.g., for machining), or handling high work loads (Fig. 42.5). However, workspace volumes tend to be smaller than those of serial or open kinematic chain robots which are comparable in size.

Still, Cartesian robots are ideally suited for covering large workspaces. Apart from the traditional design using three orthogonal translational axes, the company Güdel introduced a curved-track gantry in 1998. The concept allows one or more robot arms to track curves and to circulate in a closed transfer system. Thus, the workspace of robot can be immensely increased at high speed and precision. This may be particularly valuable in logistics or machine tending [42.12].

Two-handed dexterous manipulation can be critical for complex assembly tasks, simultaneous handling and processing of workpieces, or the handling of large objects. The first commercial robot for synchronized, two-handed manipulation was introduced by MOTOMAN in 2005 [42.13]. As a dual-arm robot imi-

tates the reach and dexterity of human arms, it can be placed on a site that previously accommodated human workers. Thus, capital expenditure can be reduced. It features 13 axes of motion: six axes per arm, plus a single axis for the base rotation (Fig. 42.6).

In parallel to industrial robots *automated guided vehicles* (AGVs) have emerged. These mobile robots are used for moving workpieces or loading equipment from point to point. Within the concept of *automated flexible manufacturing systems* (FMS) AGVs have become an important part of their routing flexibility. Initially AGVs relied on prepared floors such as embedded wires or magnets for motion guidance. Meanwhile, freely navigating AGVs are used in large-scale manufacturing and logistics. Usually, their navigation is based on laser scanners that provide an accurate two-dimensional map of the actual environment for self-localization and obstacle avoidance [42.14]. Early on, combinations of AGVs and robot arms were realized to automatically load and unload machine tools (Fig. 42.7). Only in some selected environments such as (un-)loading process equipment in the semiconductor industry these mobile arms were economically advantageous.

By 2007, the evolution of industrial robots was marked by the following main trends [42.15]:

- The average robot unit price fell to about one-third of its equivalent price in 1990, which meant that automation is becoming more affordable. At the same time, robot performance parameters such as speed, load capacity, and mean time between failures (MTBF) have dramatically improved. These improvements provide a faster return on investment, particularly for small, short-run batch production.
- Off-the-shelf components from personal computer (PC) technologies, consumer software, and the IT industry have contributed to improved performance-to-cost ratios. Today, most manufacturers integrate PC-based processors in their controllers as well as software for programming, communication, simulation, and maintenance from high-volume IT-markets.
- Multiple robots can be programmed and synchronized in real time by one controller, which allows robots to cooperate precisely on a single workpiece.
- Increasingly, vision systems for object identification, localization, and quality control have become an integral part of the robot controller.
- Robots are networked by fieldbuses or ethernet for control, configuration, and maintenance.
- New financing arrangements allow end-users to rent a robot or even have a robot workcell operated by a specialized company or even the robot supplier in order to reduce risks or to save on investment capital.
- Training and education have become important services to end-users to increase acceptance of robot technology. Specific multimedia material and courses aim at educating industrial engineers and workforce to effectively plan, program, operate, and maintain industrial robot workcells.

42.2 Typical Applications and Robot Configurations

42.2.1 Welding

Welding ranks among the most important joining processes in manufacturing. Manual welding requires highly skilled workers, as small imperfections in the weld can lead to severe consequences. Why is a robot suited to perform this critical job? Modern welding robots have the following characteristics:

- Computer control allows the programming of task sequences, robot motions, external actuators, sensors, and communication with external devices.
- Free definition and parameterization of robot positions/orientations, reference frames, and trajectories.
- High repeatability and positioning accuracy of trajectories. Typically repeatability is some ± 0.1 mm and positioning accuracy is of the order of ± 1.0 mm.
- High speeds of the end-effector of up to 8 m/s.
- Typically, articulated robots have six DOF so that commanded orientations and positions in their workspace can be reached. Workspace extensions by mounting the robot on a linear axis (seventh DOF) are common, especially for welding of large structures.
- Typical payloads of 6–100 kg.
- Advanced programmable logic controller (PLC) capabilities such as fast input/output control and synchronizing actions within the robot workcell.
- Interfacing to high-level factory control through fieldbuses or ethernet.

Metal inert/active gas (MIG/MAG) welding is the predominant application of industrial robotics today. The automatic arc-welding process is based on a consumable wire electrode and a shielding gas that are fed through a welding gun (Fig. 42.8). Electric current sources, torches, and peripheral devices for automatic cleaning and maintaining the torch (anti-splatter, wire-cutting, tool changer etc.) are offered by specialized companies. Often sensors are used to track welding gaps and measure weld seams either before or synchronously to the welding process, thus adapting the robot's trajectory in the presence of workpiece variation and distortion. Also, cooperating robots have been introduced where one robot fixes and moves the workpiece in synchronization with another robot carrying a welding tool so that the weld can be performed with the pool of molten metal horizontal.

Another interesting robot task is multilayer welding. In regular intervals the robot measures the profile of the weld gap and adaptively generates subsequent tool paths to apply successive weld seams until the final required geometry is reached. In the example shown in Fig. 42.8c up to 70 seams can be produced by the robot to weld thick metal plates.

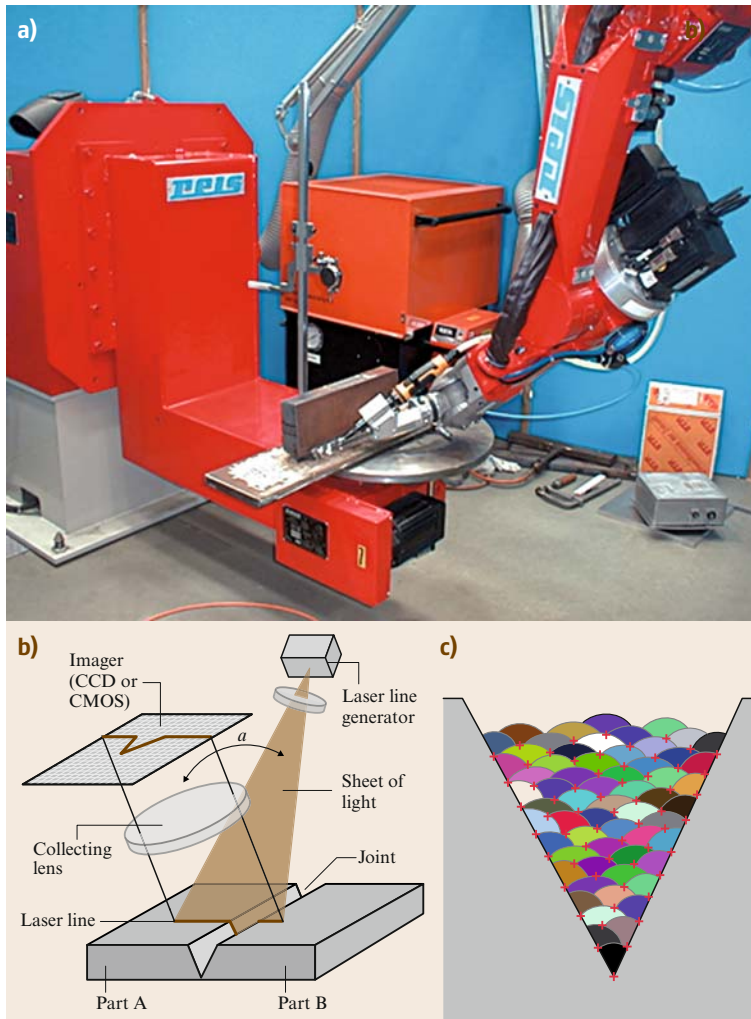


Fig. 42.8a–c Robot welding with sensor guidance. **(a)** The welding robot workcell consists of a 6 DOF robot, a turning table so that the seam is accessible and welded in a flat position, the weld source with a fume extractor and the welding torch. Safety fences and light shields are not shown. The welding torch is attached to the robot flange. A service station (not shown) in the robot workcell is approached regularly to clean the gas nozzle. **(b)** A typical weld seam sensor is based on the so-called laser triangulation principle. The sensor is attached to the robot and a laser projects a sharp stripe on the seam. The stripe is detected by an imager, typically a 2-D CCD camera. From the extracted contour the position of the seam relative to the sensor is calculated. **(c)** Multiple seams can be generated based on this information

42.2.2 Car Body Assembly

Early on, car body assembly became the predominant robot application as the benefit for robot automation was apparent. Handling and positioning the metal sheets, spot welding, and transport of the body frames was either hazardous, physically demanding to the worker, or difficult to realize on fixed automation lines given the desired variety of car body configurations to be assembled on the one production line (Fig. 42.9). In the stamping section metal sheets are cut into plates (or *blanks*) ready for pressing into body panels. In subsequent steps robots load the panels onto a tray that fixes the panels for other robots to be spot-welded. After inspection the finished bodies are transferred by conveyor to the paint shop.

Individual assembly units and components such as motors, transmissions, axles, doors or fenders are pre-mounted in separate areas. The car body is delivered to the body assembly area *just in sequence*, i. e. at the right time and place on the assembly line. The climax of the assembly process (wedding) – when the engine, drive and chassis first meet takes place before the last parts are mounted on the car.

Today's industrial robots, particularly in the workload category of 100–300 kg, are to a large extent the results of the requirements stemming from this application:

- Required repeatability of at least ± 0.5 mm under typical loads of some 100 kg for the spot-welding

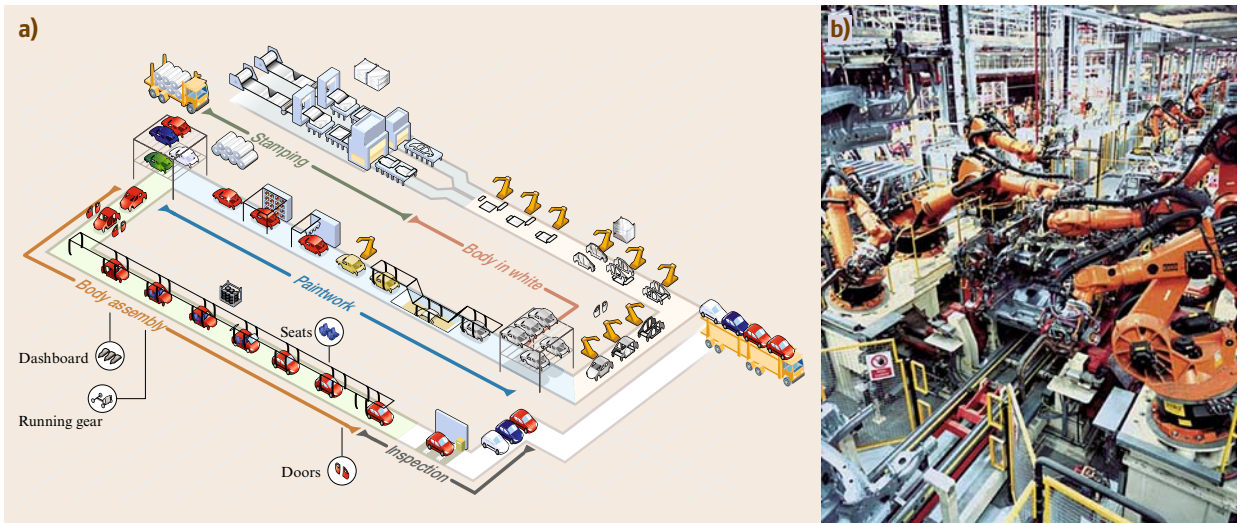


Fig. 42.9a,b Car body assembly. **(a)** A car body assembly usually follows the illustrated steps: Stamping of the metal sheet into plates, fixing and alignment of the plates on trays, spot welding, painting the car body, and final assembly of the car body (doors, dashboard, windcreens, power-train seats, and tires). Car factories can host well over 1000 robots working two to three shifts per day. (Courtesy PSA Peugeot Citroën, Paris and Art Movies, Paris) **(b)** The Mercedes A class assembly in Rastatt Germany is highly automated. The picture shows spot welding robots along the, *body in white* transfer line. Trays carrying car bodies pass through the *robot garden*

tool and cable package leads to stiff and heavy arm structures. A typical robot weight-to-payload ratio is of the order of 1500 kg:150 kg.

- Three-shift continuous operation requires highest reliability of both robot and equipment. Typical mean time between failure (MTBF) is reported to be around than 50 000 h [42.1].
- The line capacity depends on the robot's speed to place spot welds. Thus, the *point-to-point* (PTP) movement time between welding positions should be kept as short as possible. This is mainly achieved by powerful drives so that the mean power consumption of the robot drives is typically 5 kW.
- Most of the trajectories, positions, and orientations are generated using *offline programming* (OLP) systems. Accurate simulations of the robot motion depend on robot models, which incorporate both robot kinematic properties and controller characteristics. The *realistic robot simulation* (RSS) interface is the accepted standard format for robot models of offline programming systems [42.16].

42.2.3 Painting

Hazardous working conditions for human operators motivated Trallfa, a Norwegian company, to develop simple

spray-painting robots in 1969, particularly for spraying bumpers and other plastic parts in the automotive industry. Initially pneumatically driven for antiexplosion reasons, today's robot designs are fully electric with greatly improved spray guns. They also have hooks and grippers to open hoods and doors during the painting task. Hollow wrists that house gas and paint hoses allow fast and agile motions. Spray guns for robots have evolved dramatically for delivering uniform quality using as little paint and solvent as possible and also for switching between different paint colors. Originally spray-painting robots replicated movements copied from human workers. Most of the programming for robot painting today is done offline as state-of-the-art programming systems offer integrated process simulations to optimize paint deposition, thickness, and coverage (Fig. 42.10).

42.2.4 Material Transfer Automation

Generally, industrial practice in robot workcell planning aims at finding a compromise between reducing the variation of the workpiece position and the cost of sensor systems to compensate for residual variation. Nearly all parts arrive at robot workcells in a repeatable manner, either being stored in special magazines, or by being

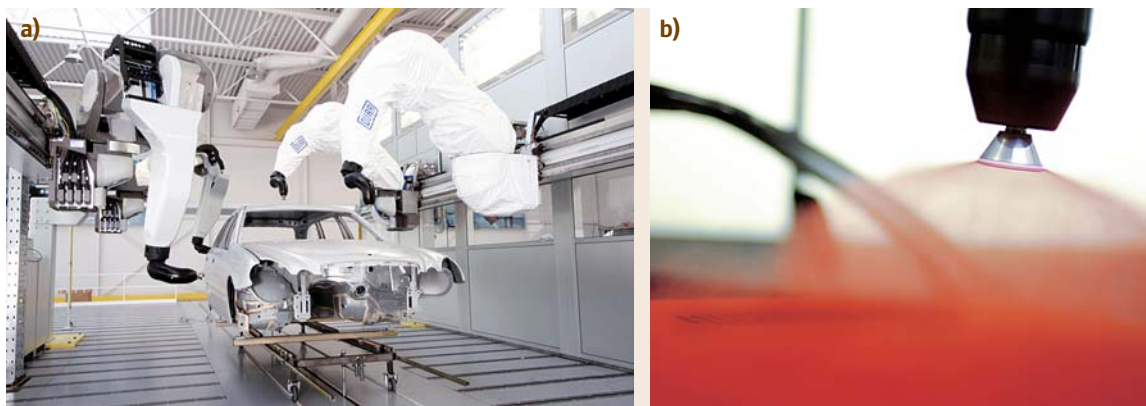


Fig. 42.10a,b Painting robots. (a) A multi-robot workcell for car body painting (b) High-speed rotating atomizer and charge. Painting robots. (a) are used for surface coating of car bodies and other parts. All current paint materials such as solvent-based paints, water-borne paints and powder can be applied. In car production, multiple robots work in parallel for optimal throughput and accessibility of the car body. Most of the programming which includes the synchronisation of the robots is performed offline. Paint guns are critical to the process quality. Figure (b) shows a Dürr EcoBell2 paint gun which atomizes the paint material at the edge of the rotating bell disk by centrifugal forces (by courtesy Dürr, Stuttgart)

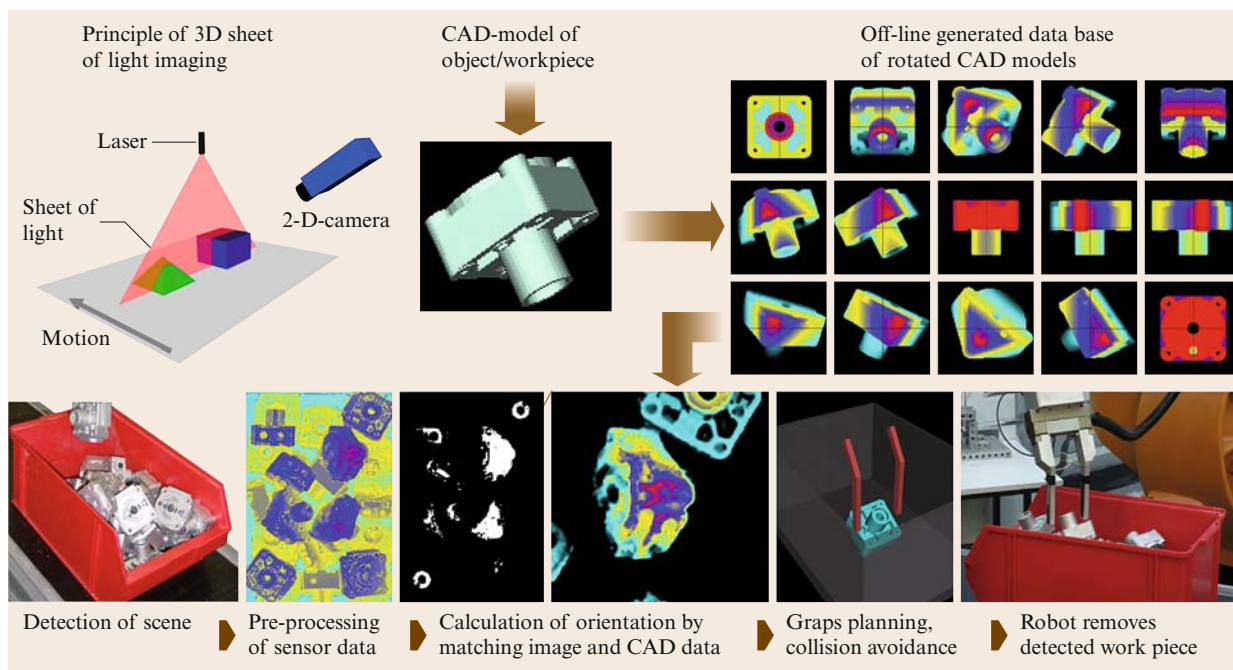


Fig. 42.11 Robot bin picking. A scene containing the objects is acquired by a 3-D sensor e.g. based on the laser triangulation principle. Beforehand this approach turns the CAD object virtually in discrete spatial angles in an off-line calculation. Feature histograms for each view are generated and stored in a database. A best match between actual feature histograms and the simulated sets of histograms determines the location of the object. A grasp has to be selected and a collision-free trajectory is generated. A typical cycle time of a location process is between 1 and 2 s

transported by vibrating devices that allow the parts to settle into a predictable orientation.

If randomly oriented in a carrier or a box, parts have to be properly located so that the robot can grasp them appropriately. This challenge in industrial robotics has been referred to as *bin-picking* and has been investigated by numerous researchers since the mid-1980s [42.17]. Even though an abundance of approaches has been presented a cost-effective standard solution has not been established yet [42.18].

Barely 10% of industrial robot installations in 2006 were sensor-equipped. However, it is expected that many

future robots will have standard embedded force-torque and vision sensors. This is a precondition for advanced vision for object identification and localization in manufacturing.

One method for determining random object locations using the object's computer-aided design (CAD) data starts with point clouds of a scene acquired by a three-dimensional (3-D) sensor (Fig. 42.11). A selection of 3-D sensors is described in Chap. 22. The CAD model has been virtually turned into discrete spatial angles in an offline calculation. Feature histograms for each view are generated and stored in a database with

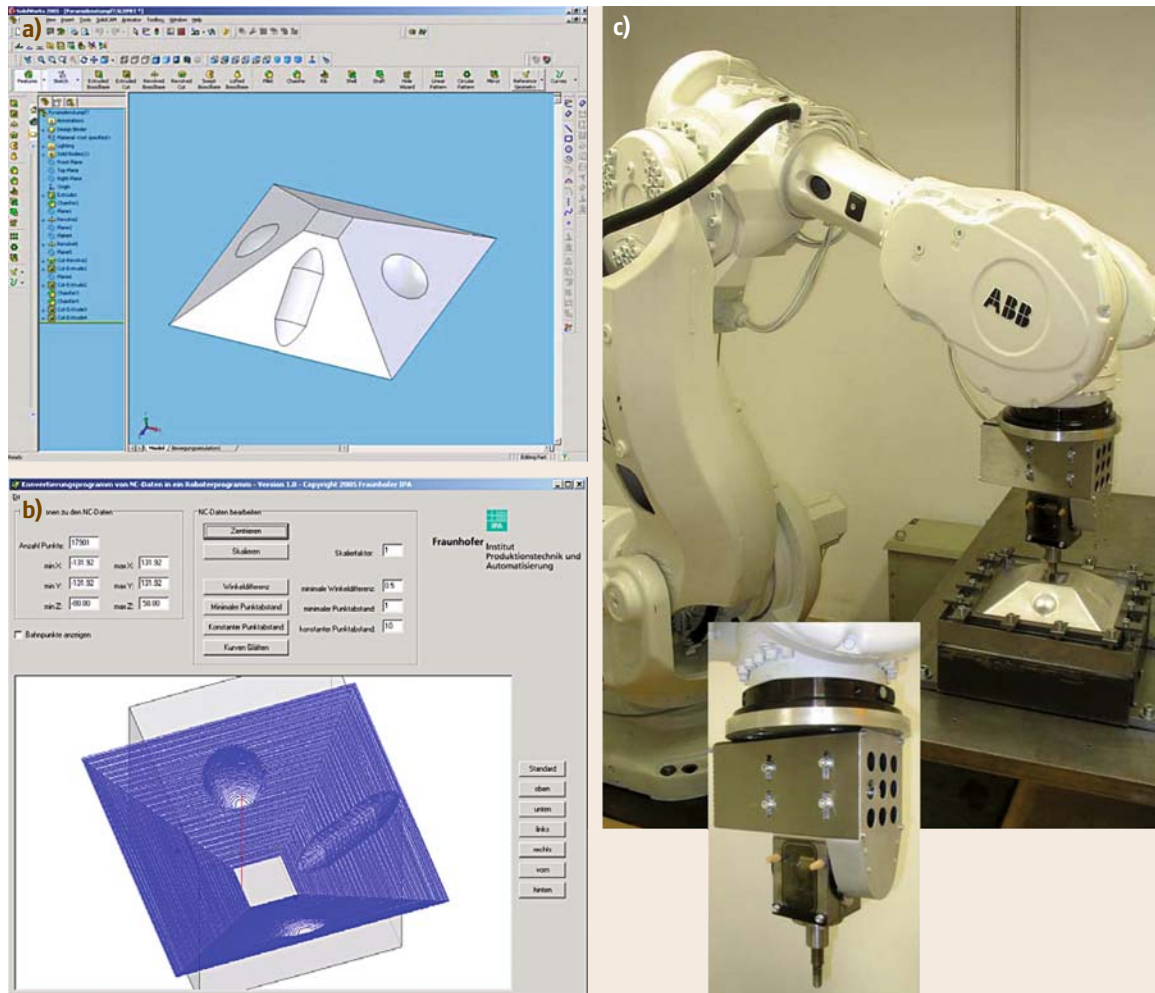


Fig. 42.12a–c Trajectory generation for incremental machining processes. In this example, the forming process of metal sheets is based on an oscillating stamp (amplitude 1 mm, 50 Hz frequency) which locally plastifies the metal in incremental steps. From the CAD model (a), the robot's trajectories are calculated offline on the basis of specific material models (b). Each line represents a part of the tool trajectory. (c) Shows a demonstrator robot workcell with the tool in detail

the given angular information (typically in 10° steps for all spatial angles).

A best-match process compares actual feature histograms with the simulated sets of histograms in the database. For the determined location of the object, a grasp has to be selected and a collision-free trajectory be generated. The latter steps can be quite critical as residual workpieces at the box bottom may constrain the robot's grasp and departing trajectory [42.19].

42.2.5 Machining

Compared to a milling machine or a lathe, standard robots possess much less stiffness (by a factor of 20–50 times), but much greater dexterity. A serial robot's stiffness is usually very anisotropic throughout its working space and may vary for a typical heavy duty model in the range 200–700 N/mm. Therefore, robots can machine workpieces (grinding, fettling, polishing etc.) provided that tool forces can be reduced to acceptable values for a given robot manipulator. This incremental approach to machining, particularly for cutting and forming, can produce good results. However, these sequential robot motions have to be generated automatically, which requires merging the process information with the workpiece geometry.

An example of a novel process that benefits from the robot's versatility in terms of programming, dexterity, and cost is the so-called incremental forming process of metal sheets. Without using any costly form, metal

sheets are clamped in a rigid frame and the robot produces a given 3-D contour by guiding a tool equipped with a high-frequency oscillating stamp (amplitude typically 1 mm at a frequency of 50 Hz) over the metal surface. The robot's trajectories are calculated from the CAD model on the basis of specific material models. The robot's program is generated and passed to the controller. One-off housings for machines, prototype panels or customized car panels can be economically produced using this method [42.20]. Figure 42.12 depicts the sequence of actions for automatically generating and executing the forming program.

42.2.6 Human–Robot Cooperation for Handling Tasks

Robots for human augmentation (force or precision augmentation) stretch from fully automated operation to acting as a servo-controlled balancer (see also Chap. 57). As an example: in a car drive train assembly the heavy gear box is grasped by the robot which balances it softly so the worker can insert it precisely into the housing (Fig. 42.13). Preprogrammed virtual walls give the worker a realistic feeling of constraints.

The central interface for the worker's tactile commands is a handle equipped with safety switches. These switches trigger the force-torque sensor that is attached to the robot's flange. Upon touching both safety switches (two-handed operation) the force-torque sensor is activated and the robot control is set to a safe



Fig. 42.13 Human-robot-cooperation for handling tasks. Inside a regular workcell which is secured by light curtains, the robot handles gear boxes at regular speed in fully automated mode. Upon approaching the light curtain at reduced speed, the worker grasps the safety switch which activates both the reduced-speed mode and the force-torque sensor. The worker guides the robot almost effortlessly by its handle so that the gear-box is balanced with precision into the rear axle frame for final assembly (Fraunhofer IPA, Stuttgart)

reduced end-effector speed (of some 50 mm/s). Thus, the sensor scales the applied force/torque information into a compliant robot motion. Obviously the physical human–robot cooperation has to obey safety precautions as the robot’s and worker’s workspaces overlap.

The ISO 10218-1:2006 standard specifies requirements and guidelines for the inherent safe design, protective measures, and information for use of industrial robots. It describes basic hazards associated with robots, and provides requirements to eliminate or adequately reduce the risks associated with these hazards. A novel element of this revised standard is the regulation of so-called collaborative operation, where the robot works in direct cooperation with a human within a defined workspace. Basically the collaborative operation depends on several criteria, which have to be met by the robot workcell [42.21]:

1. The hand-guiding equipment shall be located in the area of the end-effector.
2. The robot moves with safe reduced speed (less than 250 mm/s) and safe monitored position. This position monitoring shall be according to at least category 3 of ISO 13849, unless a risk assessment is performed and determines that a higher category is required [42.22, 23].
3. The robot must sense and keep a safe distance from the human. The distance relates to the attended

speed. The distance and speed monitoring shall be according to at least category 3 of ISO 13849, unless a risk assessment is performed and determines that another category is required.

If a robot’s total power consumption can safely be limited to 80 W as well as its static impact forces to 150 N no additional sensor-based safety precautions are needed. Again, these conditions have to be secured by a risk assessment or security systems which comply with at least category 3 of ISO 13849 [42.21]. Currently, novel control algorithms, kinematic and actuator designs are being investigated to realize so-called intrinsically safe robots [42.24].

The described workcell complies with the ISO standard in such a way that the presence of the human is detected by activating both safety switches of the handle (first condition). The robot’s built-in safe controller safely measures the end-effector’s position and limits its velocity [42.25] (second condition). Meanwhile most of the robot manufactures provide safe category 3 controllers. If the worker’s presence in the workspace is not known (third condition, not applicable here) a safe sensor system has to detect safely the workers location according to category 3. Three-dimensional sensors meeting category 3 first appeared on the market in 2006, thus opening up a wide field of potential collaborative operations [42.26].

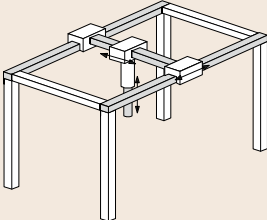
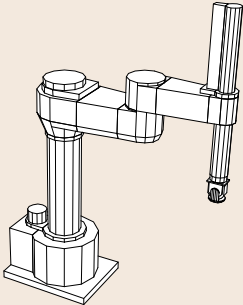
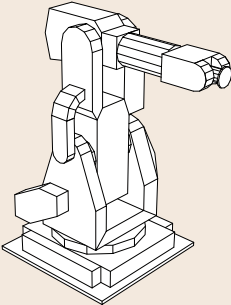
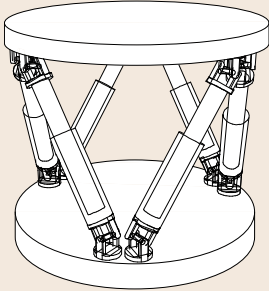
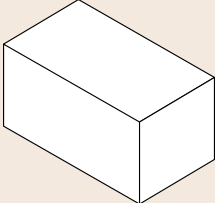
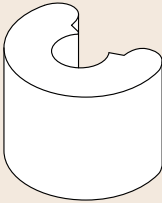
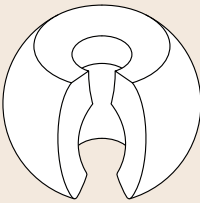




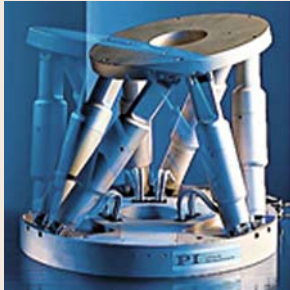
42.3 Kinematics and Mechanisms

The choice of mechanism, its kinematic properties, the computation methods used to determine joint motions, and the intended application of a robot manipulator are all closely related. The diagrams in Table 42.1 show several common types of robot mechanism.

With advances in the state of the art in kinematic algorithms and computer hardware processing capabilities, computation is much less of a constraint on mechanism choice than it was for early robot designers. The choice of mechanical structure of the robot depends mostly on fundamental mechanical requirements such as payload and workspace size. Considering a given level of cost, there is usually a tradeoff between workspace size and stiffness. To enable the robot to reach inside or around obstacles it is clearly advantageous to use an articulated mechanical design.

Considering also the stiffness and accuracy (in a practical sense considering what is reasonable to build), the picture is more complex. Each of the first three types in Table 42.1 we refer to as *serial kinematic machines* (SKMs), while the last is a *parallel kinematic machine* (PKM). To obtain maximum stiffness, again for a certain minimum level of cost, the end-effector is better supported from different directions, and here the PKM has significant advantages. On the other hand, if high stiffness (but not low weight and high dexterity) is the main concern, a typical computerized numerical control (CNC) machine (e.g., for milling) is identical in principle to the gantry mechanism. There are also modular systems with servo-controlled actuators that can be used to build both robots with purpose-designed mechanisms.

Table 42.1 Mechanical types of motions with different kinematic: Gantry is what a Cartesian coordinate robot is typically called, the SCARA is a *selective compliant articulated robot* (or *selective compliance assembly robot arm*, since the main application is planar assembly), the *Articulated* (all joints are rotary and placed in series after each other) robot is often referred to as an arm robot, and the *Parallel* is shown with prismatic joints, but can also have revolute joints such as the Delta robot (Fig. 42.5). Combinations are common

Gantry	SCARA	Serial articulated	Parallel
			
Form of workspace			
			
Product example			
			

42.4 Task Descriptions – Teaching and Programming

We cannot instruct a robot in the same way that one would instruct a human worker how to carry out a task. With skilled workers knowing the applications, devices, processes, and the general requirements on the product to be manufactured, we would only need to summarize *what* needs to be done.

In practice, since it is difficult to encode much of the required background knowledge, we aim for programming principles that resemble instructing a (totally) unskilled worker, telling precisely *how* every aspect of the task is to be performed. That requires a much more explicit way of instructing the robot, but one which is

still human-friendly. More specifically, as a wish-list that we will come back to at the end of the chapter, we would like to teach robots by

- manually guiding the robot to the positions of interest, or even along the desired paths or trajectories if human accuracy is enough
- having simple ways to make use of CAD data whenever available
- using different complementary modalities (paths of communication between the human and the robot), such as speech and gestures
- choreographing the task movements, for instance loops and conditions, without requiring extensive programming competencies
- means of describing acceptable variation, e.g., as expected or normal deviations from the nominal path
- specification of how external sensing should be used for new types of motions or for handling unknown variations

Initially, mainly during the 1970s and 1980s, there were some painting robots that could be programmed by manual guidance. This was possible due to the following abilities

- Applications such as painting permit the use of a lightweight arm, including the end-effector, possible balanced with respect to gravity if needed.
- The accuracy requirements were (compared to today) modest so it was possible to use back-drivable drive trains and actuators, and the definition of the motions could then be done manually by the operator moving the end-effector along the path. The recorded poses, including the timing/speed information, then define the programmed motion.
- Since no inverse kinematics was needed during programming or real-time operation, it was not a problem from a computing point of view to use arm kinematics without singularities in the workspace.
- The requirements of optimality of painting motions were also modest, compared to recent years when environmental conditions (for nature and workers) calls for minimal use of paint.

It is often referred to as an inevitable problem that there are singularities to be handled within the workspace. However, to simplify the kinematics and its inverse from a software point of view (e.g., during the 1980s considering the power of the microprocessors and algorithms available at that time), robots were actually designed to have simple (to compute) inverse kinematics. For instance, the wrist orientation was decoupled

from the translation by the arm by using wrist axes that intersected with the arm axes. The resulting singularities within the workspace could be managed by restrictions on wrist orientations, but an unfortunate implication was that robot arms were no longer back-drivable (close to the singularities) when designs were (due to engineering and repeatability requirements) adopted to standard industrial controllers. Then with the development of microprocessor-based industrial controllers and the definition of motions based on jogging (manual moves by for instance using a joystick) and CAD data, still lacking efficient and robust techniques for human-like instructions (speech, gestures, etc.), the means of robot programming became closer to computer programming (extended with motion primitives).

Robot programming languages and environments have traditionally been separated into online programming (using the actual robot in situ) and offline programming (using software tools without occupying the robot). With the increasing power of offline programming tools, their emerging ability to connect to the physical robot, and the increasing level of software functions that are embedded into the robot control system, online programming is now unusual, except to verify and manually adjust programs generated offline. Still, of course, it is economically important to minimize downtime for robot programming, and advanced sensor-based applications may be too hard to develop without access to the true dynamics of the physical workcell for fine-tuning. Nevertheless, robot languages and software tools must provide for both methods of programming.

Even though robot languages from different manufacturers look similar, there are many semantic differences that have to do with both the meaning when programs are running (the robot performing its operations) and the way the robot is instructed. The need to ensure that existing robot programs can operate with replacement robots and controllers, and also to make use of existing knowledge in robot programmers and incorporated into offline programming software, requires manufacturers to continue to support their original proprietary languages. Features such as backward execution (at least of motion statements) and interactive editing during interpretation by the robot (in combination with restrictions to make the programming simpler) also make robot languages different from conventional computer programming languages.

During the last decade and in current developments, the trend has been to focus on the tool (robot end-effector) and on the process knowledge needed for the manufacturing process, and to let the operator express

Table 42.2 Example: Simple pick-and-place operation with a typical industrial robot controller

ABB Rapid (a proprietary language) program: positions obtained from a charge-coupled device (CCD) camera; end-effector tool is pneumatic. MoveJ denotes joint-space motion and MoveL denotes linear/Cartesian motion. Basically there are four arguments for such motions: target position (here relative, by use of the offset function), the maximum motion speed (here using predefined v-constants with the number specifying the desired speed in mm/s along the path), the desired accuracy before continuing (fine referring to no corner blending at all), and the tool being used (including frames, inertia, etc., here in the tool0 data record).	
PROC cam_pick()	
MoveJ app_point, v1500, z25, tool0;	// Preposition near the working table (using a 25 mm tolerance)
MoveJ Offs(camera1 l,x,y,-30), v1500, fine, tool0;	// Approach 30 mm above the object object, position x, y from webcam
temp1:=CRobT();	// Acquire current position
MoveL Offs(temp1,0,0,-38), v400, fine, tool0;	// Move to object: account for suction cup flexibility (8 mm)
Set DO08;	// Vacuum ON, pick object
WaitTime 1.5;	// Wait 1.5 s
MoveL Offs(temp1,0,0,10), v400, fine, tool0;	// Move up 10 mm
MoveJ app_point, v1500, z25, tool0;	// Move “fly-by” to box with 25 mm tolerance
MoveJ place_box, v1500, fine, tool0;	// Move to box position
Reset DO08;	// Vacuum OFF, release object
Set DO07;	// Blow ON
WaitTime 0.8;	// Wait 0.8 s
Reset DO07;	// Blow OFF
ENDPROC	

Table 42.3 Robot-related centricity, ranging from a high-level view of the work to be accomplished in terms of the product to manufacture, to a low-level robot motion view that, in practice, constraints what manufacturing operations can be performed

<div><div>User application</div><div>Robot constraint</div></div>	Product: Description of the final shape and assemblies of the workpieces, in terms of that product; the robot system plans the operations.
	Process: Based on known sequences of specific manufacturing operations, each of these is specified in terms of their processes parameters.
	Tool: The motion of the robot-held tool is specified in terms of programs or manual guidance; the user knows the process it accomplishes.
	Arm: The robot arm and its end-plate for tool mounting is programmed how to move in Cartesian space; the user knows the tool.
	Joint: For each specified location the joint angles are specified, so straight-line motions are difficult; the robot provides coordinated servo control.

the robot task in such terms. This development results in a need for an increased level of abstraction to simplify the programming, reflecting the fact that the so-called robot programmer knows the production process very well, but has quite limited programming skills. To understand why such a high level of abstraction has not come into widespread usage, we may compare this with

the early days of industrial robotics when there was no kinematics software built into the controllers, and hence the robots were programmed via joint-space motions. (Kinematics here deals with the relation between robot motors/joints and the end-effector motions.) Built-in inverse kinematics permits tool motion to be specified in Cartesian coordinates, which is clearly

a great simplification in many applications. That is, the robot user could focus more on the work to be carried out by the robot and less on the robot itself. However, robot properties such as joint limits cannot be neglected. Until the beginning of the 1990s, robots did not follow programmed trajectories very well at high speeds or accelerations, and full accuracy could only be achieved at low speed. Modern robot systems with high-performance model-based motion control perform their tasks with much greater accuracy at high speed due to model based control features, see Chap. 6 [42.27, 28].

There are increasing opportunities to raise the level of abstraction to simplify the use of robots even more by encoding more knowledge about the robot, tools, the process, and workcells into control systems.

Example 42.1:

A machine part consisting of plates and pipes of aluminium is to be manufactured, and there is robot equipment available for production of this (and other) types of workpieces.

- A *product*-centric system would generate configurations and robot programs, and instruct the operator for whatever manual assistance that might be needed (e.g., clamping and fixturing). The system would determine the welding data such as the type of welding and how many passes for each seam.
- A *process*-centric system would instead accept input in terms of the welding parameters to use, including the order of the welds. The system would select input signals to the process equipments (such as what output voltage the robot controller should set such that the welding will be done with a certain current), and robot programs specifying the motions of the weld pistol would be generated.
- A *tool*-centric way of programming would require the operator to set up the process manually by configuring the equipment in terms of their native settings, and appropriate tool data would be configured such that the robot controller can accomplish the programmed motions, which are specified by giving coordinates and motions data referring to the end-effector.
- An *arm*-centric system is similar to the tool-centric approach in that Cartesian and straight-line motions can be (and need to be) explicitly specified/programmed by the programmer, but extra work is needed since the robot does not support a general tool frame.
- A *joint*-centric style would be needed if one of the very early robot systems is being used, requiring

a straight line to be programmed by lots of joint-space poses close to each other.

Thus, the question is, are you programming robot joint servos to make the robot provide a service for you, are you programming/commanding an arm how to move a tool, is it the tool that is made programmable by means of the robot, is it manufacturing services that are ordered by specifying the desired process parameters, or is it an intelligent system that simply can produce your product?

As a final goal related to the product-centric view, so-called task-level programming is desirable. This has been a goal since the 1980s, and implicitly also since the very beginning of robotics. It would mean that the user simply tells the robot what should be done and the robot would know how to do it, but it would require extensive knowledge about the environment and so-called machine intelligence. The need for extensive modeling of the environment of the robot is well known. Sensing of the environment is costly, but in an industrial environment it should only be needed occasionally. Modeling has to encompass full component dynamics and the limitations of the manufacturing process. With these difficulties, task-level programming is not yet achievable in practice.

As indicated in the application examples, it is now common practice to generate robot programs from geometric data in CAD files. That is, the CAD application could be the environment used for specifying how the robot should perform the required operations on the specified parts. This is not quite task-level programming since human operators do the overall planning. CAD software packages are powerful 3-D tools and are now very common among manufacturing companies. Consequently, using those tools for robot programming is desirable since the operator may start the offline programming of the necessary manufacturing operations using the 3-D model of the product. There are two possible approaches:

1. Use the CAD interface to parameterize predefined robot programs, tailoring their behavior by using geometric information [42.16, 29]. This means defining motions, adding process parameters, deciding on what to do with data from devices, etc.
2. Generate the entire robot code from the information extracted from a CAD file, including the interface code to handle cell devices [42.30, 31].

The example presented in Fig. 42.14 was designed to run from the CAD package INVENTOR [42.32],

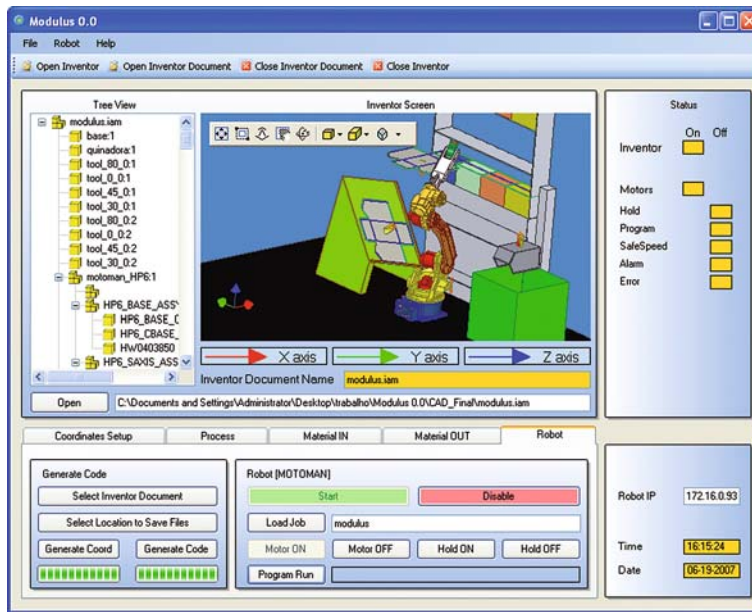


Fig. 42.14 Human-user interface for the CAD programming environment

from Autodesk. Basically it enables a user to program a workcell composed by a robot and a bending machine. The robot is used to feed and handle the sheets of metal to the bending machine, and to palletize the final product. The depicted pro-

gramming interface is an illustrative example of the possibilities, while there are a number of major vendors of CAD or offline programming system providing similar features in current and upcoming products.

42.5 End-Effectors and System Integration

It is interesting to note that connecting different workcell devices with each other, and integrating them into a working system, is hardly mentioned in the robotics literature. Nevertheless, in actual nontrivial installations, this part typically represents half of the cost of the installation. The automation scenario includes all the problems of integrating computers and their peripherals, plus additional issues that have to do with the variety of (electrically and mechanically incompatible) devices and their interaction with the physical environment (including its inaccuracies, tolerances, and unmodeled physical effects such as backlash and friction). The number of variations is enormous so it is often not possible to create reusable solutions. In total this results in a need for extensive engineering to put a robot to work. This engineering is what we call system integration.

Carrying out system integration according to current practice is not a scientific problem as such (although how to improve the situation is), but the obstacles it comprises

form a barrier to applying advanced sensor-based control for improved flexibility, as is needed in short-series production. In particular, in future types of applications using external sensing and high-performance feedback control within the workcell, system integration will be an even bigger problem since it includes tuning of the feedback too.

For long production runs, the engineering cost of system integration is less of a problem since its cost per manufactured part is small. On the other hand, the trend towards shorter series of customized products, or products with many variants that are not kept in stock, calls for high flexibility and short changeover times. Flexibility in this context refers to variable product variants, batch sizes, and process parameters. In particular this is a problem in small and medium-sized productions, but the trend is similar for larger enterprises as flexibility requirements are continuously on the rise. One may think that simply by using well standards for *in-*

Table 42.4 Stages of system integration, typically carried out in the order listed

Physical	Selecting equipment based on dimensioning for mechanical size, load, and stress
	Mechanical interfacing (locations, adapter plates, etc.)
	Electrical power supply (voltages and currents for robots, effectors, feeders, etc.)
	Connections for analog signals (shielding, scaling, currents, binary levels, etc.)
Communication	Interconnections for single-bit digital I/O
	Byte-wise data communication, including latencies and bit rates
	Transfer of byte sequences
Configuration	Configuration of messages between interacting devices
	Establishment of services
	Tuning for performance and resource utilization
Application	Definition of application-level functions/services
Task	Application programming, using the application-level services

put/output (I/O) and well-defined interfaces, integration should be just a matter of connecting things together and run the system. Let us examine why this is not the case.

Example 42.2 Step 1 of 3:

Simple pick-and-insert (assembly) operation. As a first step let us consider only one robot performing pick-and-place from/to known locations, and assume it is a well-known object such that we can use one specific type of gripper (such as a gripper, in this case based on a vacuum cup). We then only need one digital output from the robot controller. Let us call this output GRIPPER (i. e., the name of the number of the output) and the values GRASP and RELEASE for the high and low signals depending on the sign of the hardware connection. An example of a robot program can be found in the previous section.

Of course the object-oriented software solution would be to have a gripper class with operations grasp

and release. That would be appropriate for a robot simulator in pure software, but for integration of real systems such encapsulation of data (abstract data types) would introduce practical difficulties because:

- values are explicit and accomplished by external (in this case) hardware, and for testing and debugging we need to access and measure them,
- online operator interfaces permit direct manipulation of values, including reading of output values. The use of functions of object methods (so-called set:ers and get:ers) would only complicate the picture; maintaining consistency with the external devices (such as the definitions of the constants GRIPPER, GRASP, and RELEASE) would be no simpler.

Already in this toy example we can see that the user I/O configuration depends on both connected external devices (the gripper), and also on what I/O modules (typically units on some kind of field bus) that are in-

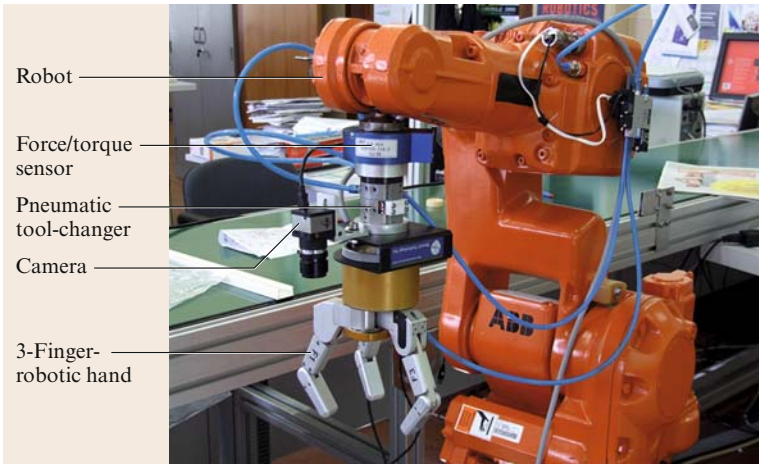


Fig. 42.15 Force controlled robot with multiple tools: force-torque sensor (JR3), robotic hand (Barrett), CCD USB camera (uEye) and tool-changer (ATI). This type of flexible gripper is not (yet) common in industry, where simple 1 DOF grippers with fingers tailored to the product are normally used

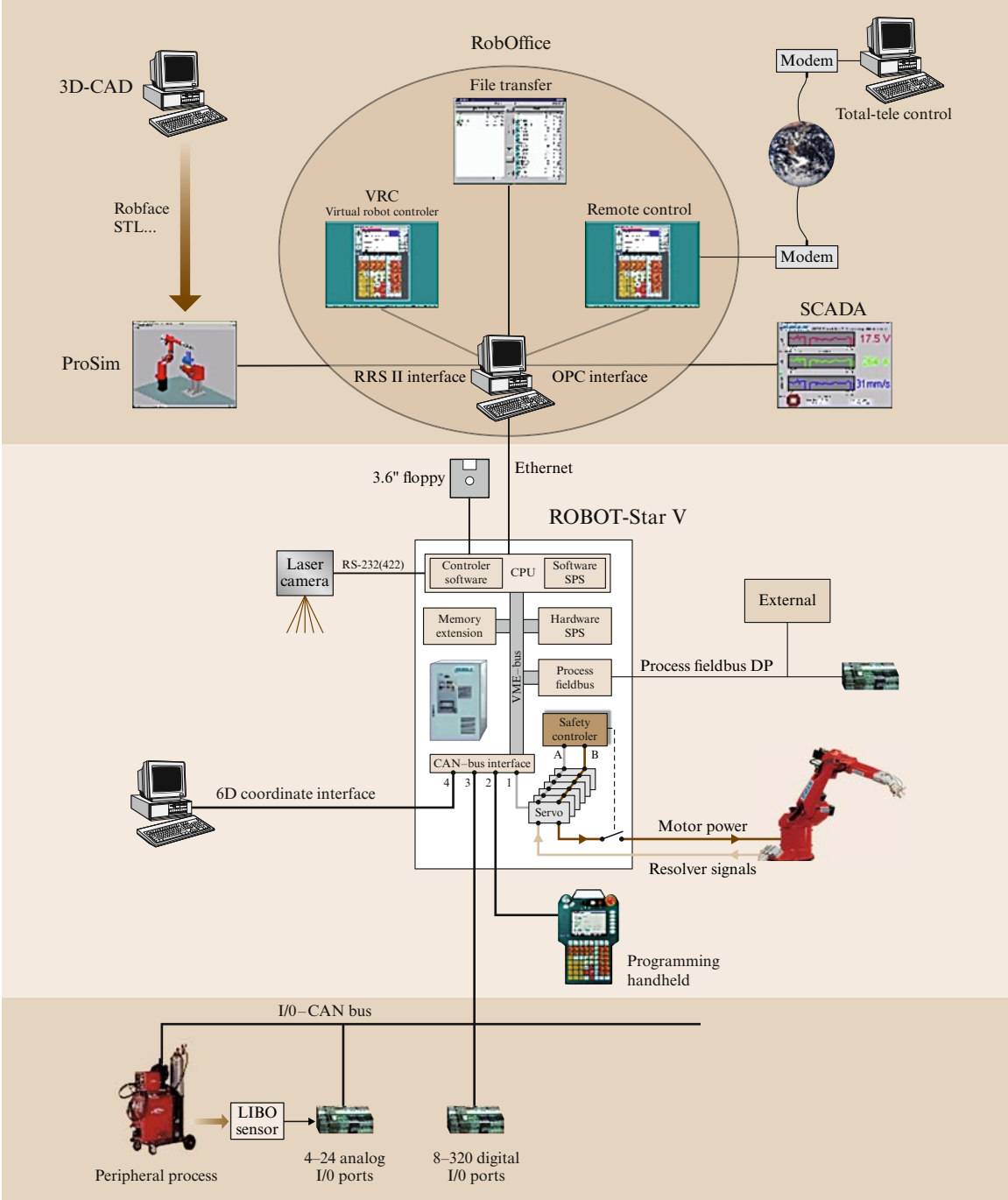


Fig. 42.16 Control modules and interfaces (Reis) on different levels, from low-level peripheral interfaces to high-level factory interfaces. The robot controller contains the middle part. The vertical integration in combination with heterogeneous hardware forms a major challenge (by courtesy Reis Robotics, Obernburg)

stalled in the robot controller (which may be subject to change when another task is given to the robot). Returning to our example, selecting modules and interfaces extends to the items of Table 42.4, where the application and task levels relate to robot programming and the research issue of how to simplify system integration, respectively.

Example 42.2 Step 2 of 3:

Simple pick-and-insert (assembly) operation. Assume we want to use a probe to detect the type and the location of the workpiece, that we need several different types of grippers depending on what object we should handle, and that the place operation needs to be a force-controlled insertion. Additionally, since we need different tools for different operations, we need a tool exchanger.

With these additional requirements we need more mechanical interfaces, the load/weight/performance considerations are more difficult (the weight of the tool exchanger and adaptor plate decreases the net payload for the grasped objects), and several more electrical interfaces need to be included.

Equipment examples are shown in Fig. 42.15, an example that extends the pick-and-insert case to include unknown positions and a gripper that can grab several types of pieces (including a wrist force sensor and force feedback in the robot fingers); details concerning robot programming and device configurations are omitted for brevity. In the presented example a three-finger robotic hand [42.33] was used to illustrate that these advanced and programmable tools are finding their way into industrial applications [42.34], for example, the hand used here has position and force-sensing capabilities. When grasping an object, the finger base and tip links move together. If a base link encounters an object it stops, while

the tip link keeps moving until it makes contact with the object as well. If the tip links encounter an object first, the whole finger assembly stops moving.

The example depicted in Fig. 42.15 shows a setup where the robot can pick the workpiece from an unknown position; the working pieces are identified using a CCD camera that returns the number of pieces and their position. The robot is then commanded to pick one and update its information from the cell [42.30, 35]. This example uses two tasks: one to receive remote commands, and the other to implement the gripper and camera services.

Example 42.2 Step 3 of 3:

Simple pick-and-insert (assembly) operation. As a final requirement, to monitor the quality of the assembly operation, assume that we want to have slip detection embedded into the gripper and that we need logging of the force control signal. The production statistics should be provided on the overall plant level of the factory.

These additional requirements call for integration of low-level devices with high-level factory control system, and is hence called *vertical integration*, whereas integration within (for instance) the workcell level is called *horizontal integration*.

Lack of self-descriptive and self-contained data descriptions that are also useful at the real-time level further increases the integration effort since data interfaces/conversions typically have to be manually written. In some cases, as illustrated in Fig. 42.16, there are powerful software tools available for the integration of the robot user level and the engineering level [42.36, 37]. The fully (on all levels) integrated and nonproprietary system, sometimes referred to as a *digital factory*, still is a challenge, particularly for small and medium-sized productions [42.38, 39].

42.6 Conclusions and Long-Term Challenges

The widespread use of robots in standard, large-scale production such as the automotive industry, where robots (even with impressive performance, quality, and semiautomatic programming) perform repetitive tasks in very well-known environments, for some time resulted in the common opinion that *industrial robotics is a solved problem*. However, these applications comprise only a minor part of the industrial work needed in any wealthy society, especially considering the number of companies

and the variety of applications. The use of robots in small and medium-sized manufacturing is still tiny.

Global prosperity and wealth requires resource-efficient and human-assistive robots. The challenges today are to recognize and overcome the barriers that are currently preventing robots from being more widely used.

Taking a closer look at the scientific and technological barriers, we find the following challenges:

- *Human-friendly task specification*, including intuitive ways of expressing permitted/normal/expected variations. That is, there are many upcoming and promising techniques for user-friendly human–robot interaction (such as speech, gestures, manual guidance, and so on), but the focus is still on specification of the nominal task (see Chaps. 58 and 59). The foreseen variations, and the unforeseen variations experienced during robot work, are more difficult to manage. When instructing a human he/she has an extensive and typically implicit knowledge about the work and the involved processes. To teach a robot, it is an issue both how to realize what the robot does not know, and how to convey the missing information efficiently.
- *Efficient mobile manipulation*. Successful implementations and systems are available for both mobility and for manipulation, but accomplished in different systems and using different types of (typically incompatible) platforms. A first step would be to accomplish mobile manipulation at all, including the *combination* of legged locomotion see Chap. 56 (for stairs and rough terrain), autonomous navigation (with adaptive but predictable understanding of constraints), dexterous manipulation, and robust force/torque interaction with environments (that have unknown stiffness). As a second step, all this needs to be done with decent performance using reasonably priced hardware, and with interfaces according to the previous item. Thus, we are far from useful mobile manipulation.
- *Low-cost components including low-cost actuation*. Actuation of high-performance robots represent about a third of the overall robot cost, and improved modularity often results in a higher total hardware cost (due to less opportunities for mechatronic optimization). On the other hand, cost-optimized (with respect to certain applications) systems result in more-specialized components and smaller volumes, with higher costs for short-series production of those components. Since future robotics and automation solutions might provide the needed cost efficiency for short-series customized components, we can interpret this as a boot-strapping problem, involving both technical and business aspects. The starting point is probably new core components that can fit into many types of systems and applications, calling for more mechatronics research and synergies with other products.
- *Composition of subsystems*. In most successful fields of engineering, the principle of superposition holds,

meaning that problems can be divided into subproblems and that the solutions can then be superimposed (added/combined) onto each other such that the total solution comprises a solution to the overall problem. These principles are of key importance in physics and mathematics, and within engineering some examples are solid-state mechanics, thermal dynamics, civil engineering, and electronics. However, there is no such thing for software, and therefore not for mechatronics (which includes software) or robotics (programmable mechatronics) either. Thus, composition of unencapsulated subsystems is costly in terms of engineering effort.

Even worse, the same applies to encapsulated software modules and subsystems. For efficiency, system interconnections should go directly to known (and hopefully standardized) interfaces, to avoid the indirections and extra load (weight, maintenance, etc.) of intermediate adapters (applying to both mechanics for end-effector mounting and to software). Interfaces can be agreed upon, but the development of new versions typically maintains backward compatibility (newer devices can be connected to old controller), while including the reuse of devices calls for mechanisms for forward compatibility (automatic upgrade based on meta information of new interfaces) to cover the case that a device is connected to a robot that is not equipped with all the legacy or vendor-specific code.

- *Embodiment of engineering and research results*. Use or deployment of new technical solutions today still starts from scratch, including analysis, understanding, implementation, testing, and so on. This is the same as for many other technical areas, but the exceptional wide variety of technologies involved with robotics and the need for flexibility and upgrading makes it especially important in this field. Embodiment into components is one approach, but knowledge can be applicable to engineering, deployment, and operation, so the representation and the principle of usage are two important issues. Improved methods are less useful if they are overly domain specific or if engineering is experienced to be significantly more complicated. Software is imperative, as well as platform and context dependent, while know-how is more declarative and symbolic. Thus, there is still a long way to go for efficient robotics engineering and reuse of know-how.
- *Open dependable systems*. Systems need to be open to permit extensions by third parties, since there is no way for system providers to foresee all upcoming

ing needs in a variety of new application areas. On the other hand, systems need to be closed such that the correctness of certain functions can be ensured. Extensive modularization in terms of hardware and supervisory software makes systems more expensive and less flexible (contrary to the needs of openness). Highly restrictive frameworks and means of programming will not be accepted for widespread use within short-time-to-market development. Most software modules do not come with formal specification, and there is less understanding of such needs. Thus, systems engineering is a key problem.

- *Sustainable manufacturing.* Manufacturing is about transformation of resources into products, and productivity (low cost and high performance) is a must.

For long-term sustainability, however, those resources in terms of materials and the like must be recycled. In most cases this can be achieved by crushing the product and sorting the materials, but in some cases disassembly and automatic sorting of specific parts are needed. There is therefore a need for robots in recycling and demanufacturing. Based on future solutions to the above items, this is then a robot application challenge.

An overall issue is how both industry and academia can combine their efforts such that sound business can be combined with scientific research so that future development overcomes the barriers that are formed by the above challenges.

References

- 42.1 The International Federation of Robotics (IFR): *World Robotics 2007. Statistics, Market Analysis, Forecasts, Case Studies and Profitability of Robot Investment* (IFR Statistical Department, Frankfurt 2007), <http://www.ifrstat.org>
- 42.2 M.P. Groover: *Automation, Production Systems, and Computer-Integrated Manufacturing*, 2nd edn. (Prentice Hall, Upper Saddle River 2000)
- 42.3 B.S. Dhillon, A.R.M. Fashandi, K.L. Liu: Robot systems reliability and safety: a review, *J. Qual. Mainten. Eng.* **8**(3), 170–212 (2002)
- 42.4 R.D. Schraft, S. Schmid, S. Thiemermann: Man-robot cooperation in a flexible assembly cell, *Assemb. Autom.* **22**(2), 136–138 (2002)
- 42.5 S.Y. Nof: *Handbook of Industrial Robotics* (Wiley, New York 1985)
- 42.6 V.D. Scheinman: Design of a Computer Controlled Manipulator. Ph.D. Thesis (Stanford University, Department of Computer Science 1969)
- 42.7 L. Westerlund: *The Extended Arm of Man. A History of the Industrial Robot* (Informations Förlaget, Stockholm 2000)
- 42.8 H. Makino: Assembly Robot, Patent 4341502 (1980)
- 42.9 G. Boothroyd, L. Altling: Design for assembly and disassembly, *Int. Inst. Prod. Eng. Res. Annal. (CIRP Annals)* **41**(2), 625–636 (1992)
- 42.10 G. Hirzinger, N. Sporer, A. Albu-Schaffer, M. Hahnle, R. Krenn, A. Pascucci, M. Schedl: DLR's torque-controlled light weight robot III – are we reaching the technological limits now?, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol. 2 (Washington 2002) pp. 170–176
- 42.11 R. Clavel: Device for the movement and positioning of an element in space, Patent 4976582 (1989)
- 42.12 R. Bloss: Innovation at IMTS, *Ind. Robot* **30**(2), 159–161 (2003)
- 42.13 Y. Kusuda: The international robot exhibition 2005 in Tokyo, *Ind. Robot* **33**(5), 342–348 (2006)
- 42.14 Y.R. Siegwart, I.R. Nourbakhsh: *Introduction to Autonomous Mobile Robots* (MIT Press, Cambridge 2004)
- 42.15 R.D. Schraft, M. Hägele, A. Breckweg: *Man and robot without separating systems*. Managers Navigator. World of Automation and Metalworking: 8th Edition for Europe (VDMA-Verlag, Frankfurt am Main 2006) pp.4–5
- 42.16 R. Bernhard, G. Schreck, C. Willnow: Development of virtual robot controllers and future trends, 6th IFAC Symp. Cost oriented Automation (Fraunhofer IPK, Berlin 2001)
- 42.17 K. Ikeuchi, B.K.P. Horn, S. Nagata: Picking up an object from a pile of objects, A.I. Memo 726, Artificial Intelligence Laboratory, Massachusetts Institute of Technology (1983)
- 42.18 J. Kirkegaard, T.B. Moeslund: Bin-picking based on harmonic shape contexts and graph-based matching, *Proc. 18th Int. Conf. Pattern Recogn. (ICPR'06)* (Hong Kong 2006) pp. 581–584
- 42.19 K. Modrich: 3D machine vision solution for bin picking applications, *Proc. Int. Robot. Vision Show (Rosemont 2007)*
- 42.20 T. Schaefer, R.D. Schraft: Incremental sheet metal forming by industrial robots, *Rapid Prototyp. J.* **11**(5), 278–286 (2005)
- 42.21 International Organization for Standardization (ISO): *Robots for Industrial Environments – Safety Requirements* (ISO, Geneva 2007), ISO 10218-1:2006/Cor 1:2007
- 42.22 International Organization for Standardization (ISO): *Safety of Machinery – Safety-Related Parts of Control Systems – Part 1: General Principles for Design* (ISO, Geneva 2006), ISO 13849-1:2006

- 42.23 International Organization for Standardization (ISO): *Safety of Machinery – Safety-Related Parts of Control Systems – Part 2: Validation* (ISO, Geneva 2003), ISO 13849-2:2003
- 42.24 A. Albu-Schäffer, C. Ott, G. Hirzinger: A unified passivity based control framework for position, torque and impedance control of flexible joint robots, *Int. J. Robot. Res.* **26**, 23–39 (2007)
- 42.25 A. Kochan: Robots and operators work hand in hand, *Ind. Robot* **33**(6), 422–424 (2006)
- 42.26 Three-dimensional control and monitoring – The first safe camera system SafetyEYE opens up new horizons for safety and security, Press release Pilz GmbH and Co. KG, Ostfildern, Germany, (2006) <http://www.pilz.de>
- 42.27 B. Siciliano, L. Villani: *Robot Force Control*, Ser. Eng. Comput. Sci. (Springer, Berlin, Heidelberg 2000)
- 42.28 J.J. Craig: *Introduction to Robotics: Mechanics and Control* (Prentice Hall, Upper Saddle River 2003)
- 42.29 U. Thomas, F.M. Wahl, J. Maass, J. Hesselbach: Towards a new concept of robot programming in high speed assembly applications, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS 2005)* (2005) pp. 3827–3833
- 42.30 J.N. Pires, A. Loureiro, G. Bolmsjö: *Welding Robots* (Springer, London 2006)
- 42.31 A. Blomdell, G. Bölsjö, T. Brogardh, P. Cederberg, M. Isaksson, R. Johansson, M. Haage, K. Nilsson, M. Olsson, T.A. Robertsson, J. Wang: Extending an industrial robot controller: implementation and applications of a fast open sensor interface, *IEEE Robot. Autom. Mag.* **12**(3), 85–94 (2005)
- 42.32 Autodesk: *Inventor Application Programming Interface. Manual* (Autodesk Inc, 2007)
- 42.33 W. Townsend: The BarrettHand grasper – programmable flexible part handling and assembly, *Ind. Robot* **27**(3), 181–188 (2000)
- 42.34 A. Wolf, R. Steinmann, H. Schunk: *Grippers in Motion* (Springer, New York, 2005)
- 42.35 J.N. Pires: *Industrial Robot Programming, Building Applications for the Factories of the Future* (Springer, New York 2007)
- 42.36 R. Zurawski: *Integration Technologies for Industrial Automated Systems* (CRC, Boca Raton 2006)
- 42.37 M. Hobday, A. Davies, A. Prencipe: Systems integration: a core capability of the modern corporation, *Ind. Corp. Change* **14**(6), 1109–1143 (2005)
- 42.38 G. Wöhlke, E. Schiller: Digital planning validation in automotive industry, *Comput. Ind.* **56**(4), 393–405 (2005)
- 42.39 E. Westkämper: Strategic development of factories under the influence of emergent technologies, *CIRP Ann. Manuf. Technol.* **56**(1), 419–422 (2007)

Underwater

43. Underwater Robotics

Gianluca Antonelli, Thor I. Fossen, Dana R. Yoerger

This chapter deals with the main underwater robotic topics. First, a brief introduction showing the constantly expanding role of marine robotics in oceanic engineering is given; this section also contains some historical backgrounds. Most of the following sections strongly overlap with the corresponding chapters presented in this handbook; hence, to avoid useless repetitions, only those aspects peculiar to the underwater environment are discussed, assuming that the reader is already familiar with concepts such as fault detection systems when discussing the corresponding underwater implementation. The modeling section is presented by focusing on a coefficient-based approach capturing the most relevant underwater dynamic effects. Two sections dealing with the description of the sensor and the actuating systems are then given. Autonomous underwater vehicles require the implementation of mission control system as well as guidance and control algorithms. Underwater localization is also discussed. Underwater manipulation is then briefly approached. Fault detection and fault tolerance, together with the

43.1 The Expanding Role of Marine Robotics in Oceanic Engineering	987
43.1.1 Historical Background	989
43.2 Underwater Robotics	989
43.2.1 Modeling	989
43.2.2 Sensor Systems	995
43.2.3 Actuating Systems	996
43.2.4 Mission Control System	998
43.2.5 Guidance and Control	998
43.2.6 Localization	1000
43.2.7 Underwater Manipulation	1001
43.2.8 Fault Detection/Tolerance	1002
43.2.9 Multiple Underwater Vehicles	1003
43.3 Applications	1003
43.4 Conclusions and Further Reading	1005
References	1005

coordination control of multiple underwater vehicles, conclude the theoretical part of the chapter. Two final sections, reporting some successful applications and discussing future perspectives, conclude the chapter.

43.1 The Expanding Role of Marine Robotics in Oceanic Engineering

The world's oceans cover 2/3 of the Earth's surface and have been critical to human welfare throughout history. As in ancient times, they enable the transport of goods between nations. Presently, the seas represent critical sources of food and other resources such as oil and gas. In the near term, we may soon see the emergence of offshore mining for metals as well as the exploitation of gas hydrates. Conversely, the ocean can also threaten human safety and damage infrastructure through natural phenomena such as hurricanes and tsunamis.

Our scientific understanding of the deep sea is expanding rapidly through the use of a variety of

technologies. The first scientific explorations were conducted primarily through the use of diving and human-occupied submersibles, complemented by a variety of other technologies such as towed or lowered instruments, trawls, dredges, autonomous seafloor instruments, and deep-sea drilling. More recently remotely operated and autonomous vehicles have begun to revolutionize seafloor exploration, often returning superior data at reduced costs. In the near future, seafloor observatories linked by fiber-optic cables and satellites will return massive amounts of data from coastal and deep-sea sites. These observations will complement those



Fig. 43.1 The ROV Jason 2 (courtesy of the Woods Hole Oceanographic Institute, <http://www.whoi.edu>)

from conventional expeditionary investigations, and will require teleoperated or robotic intervention during installation and for service. An example of a remotely operated vehicle developed for the scientific study of the seafloor is the Jason 2 vehicle developed at the Woods Hole Oceanographic Institution (shown in Fig. 43.1), and a list of remotely operated vehicles for scientific exploration appears in Table 43.1 (the last vehicle in the table, Kaiko, was lost several years ago).

Table 43.1 ROVs for scientific use

Vehicle	Depth (m)	Institution	Manufacturer
Hyperdolphin	3000	JAMSTEC ^a	ISE
Dolphin 3K	3000	JAMSTEC	JAMSTEC
Quest	4000	MARUM ^b	Shilling
Tiburon	4000	MBARI ^c	MBARI
ROPOS	5000	CSSF ^d	ISE
Victor	6000	IFREMER ^e	IFREMER
Jason	6500	WHOI ^f	WHOI
ISIS	6500	NOC ^g	WHOI
UROV 7K	7000	JAMSTEC	JAMSTEC
Kaiko	11000	JAMSTEC	JAMSTEC

^a Japan Marine. Science and Technology Center
^b Zentrum für Marine Umweltwissenschaften
^c Monterey Bay Aquarium Research Institute
^d Canadian Scientific Submersible Facility
^e Institut français de recherche pour l'exploitation de la mer
^f Woods Hole Oceanographic Institution
^g National Oceanography Centre

Offshore oil and gas installations are presently serviced almost exclusively by remotely operated vehicles (ROVs), physically connected via a tether to receive power and data, with human divers used only for the shallowest installations. Subsea systems require extensive work capability during installation, and need frequent inspection and intervention to support drilling operations, actuate valves, repair or replace subsea components, and to accomplish a variety of tasks required to maintain production rates and product quality. The trend toward robotic and teleoperated subsea intervention is certain to continue as offshore oil and gas production moves into deeper waters, and economic considerations push key production steps from surface platforms to the seafloor. Remotely operated manipulators enable these systems to perform complex tasks such as debris removal, cleaning using abrasive tools, and to operate a variety of nondestructive testing tools. The effectiveness of using ROVs decreases with depth mainly due to the cost increase and the difficulties of handling the long tether.

Autonomous underwater vehicles (AUVs) are free-swimming unoccupied underwater vehicles that can overcome the limitations imposed by ROV tethers for some tasks. Such vehicles carry their own energy supplies (presently batteries, perhaps fuel cells in the future) and communicate only through acoustics and perhaps optical links in the near future. Limited communications require these vehicles to operate independently of continuous human control, in many cases the vehicles operate completely autonomously. AUVs are currently used for scientific survey tasks, oceanographic sampling, underwater archeology and under-ice survey. Military applications, such as mine detection and landing site survey, are presently operational, and more ambitious applications such as long-term undersea surveillance are in engineering development. Presently, AUVs are incapable of sampling or manipulations tasks like those done routinely by ROVs, as typical work environments tend to be complex and challenging even to skilled human pilots.

Today, approximately 200 AUVs are operational, many of them experimental. However, they are maturing rapidly. Recently several companies now offer commercial services with AUVs. As an example, for the oil and gas industry the cost reduction of a survey performed with an AUVs instead of a towed vehicle is up to 30% and the data quality is generally higher. Likewise, commercial manufacturers in several countries now offer turnkey AUV systems for specific, well-defined tasks. Currently, remotely operated manipulators are standard

equipment for most ROVs, while on the contrary autonomous manipulation is still a research challenge; the two projects SAUVIM [43.1] and ALIVE [43.2] were devoted to studying this control problem.

43.1.1 Historical Background

Boats have been used by humans since the start of recorded history, but vehicles able to go under water are more recent. Perhaps the first recorded idea of an underwater machine came from Aristotle; according to legend he built the: *skaphe andros* (boat-man) that allowed Alexander the Great (Alexander III of Macedonia, 356–323 BC) to stay submerged for at least half a day during the war of Tiro in 325 BC. This is probably unrealistic; if true it would precede Archimedes' law, which was first articulated in approximately 250 BC. Leonardo Da Vinci may have been the first to design an underwater vehicle. His efforts were recorded in the *Codice Atlantico* (*Codex Atlanticus*), written between 1480 and 1518. Legends say that Leonardo worked on the idea of an underwater military machine but he de-

stroyed the results as he judged them to be too dangerous. The first use of feedback theory for marine control was probably the Northseeking device, patented in 1908, that used gyroscopic principals to develop the first autopilot [43.3]. From that point on, the use of feedback theory in marine control grew continuously; it is interesting to notice that the proportional–integral–derivative (PID) control commonly used today in numerous industrial applications was first formally analyzed in 1929 by Minorsky [43.4]. The first remotely operated underwater vehicle, POODLE, was built in 1953, and the ROV evolved through the 1960s and 1970s, mostly for military purposes. In the 1980s ROVs became established for use in the commercial offshore industry and began to emerge for scientific applications. The first tetherless, autonomous vehicles were built for experimental purposes in the 1970s. Currently, AUVs are becoming increasingly commonplace for scientific, military, and commercial applications. Turnkey AUV systems for a range of tasks are available from commercial vendors, and AUV services can be acquired from a number of companies [43.5].

43.2 Underwater Robotics

43.2.1 Modeling

A rigid body is completely described by its position and orientation with respect to a reference frame Σ_i , $O_i - xyz$ that is supposed to be Earth-fixed and inertial. Let us define $\eta_1 \in \mathbb{R}^3$ as

$$\eta_1 = (x \ y \ z)^\top,$$

the vector of the body position coordinates in an Earth-fixed reference frame. The vector $\dot{\eta}_1$ is the corresponding time derivative (expressed in the Earth-fixed frame). If one defines

$$v_1 = (u \ v \ w)^\top$$

as the linear velocity of the origin of the body-fixed frame Σ_b , $O_b - x_b y_b z_b$ with respect to the origin of the Earth-fixed frame expressed in the body-fixed frame (from now on: body-fixed linear velocity) the following relation between the defined linear velocities holds

$$v_1 = R_1^B \dot{\eta}_1, \quad (43.1)$$

where R_1^B is the rotation matrix expressing the transformation from the inertial frame to the body-fixed frame.

Let us define $\eta_2 \in \mathbb{R}^3$ as

$$\eta_2 = (\phi \ \theta \ \psi)^\top$$

the vector of body Euler angle coordinates in a Earth-fixed reference frame. In the nautical field those are commonly named roll, pitch, and yaw. Yaw is defined as rotation around the z axis of the fixed frame; pitch is defined as rotation around the y axis resulting after the yaw movement; and roll is defined as rotation around the x axis resulting after both yaw and pitch movements. The vector $\dot{\eta}_2$ is the corresponding time derivative (expressed in the inertial frame). Let us define

$$v_2 = (p \ q \ r)^\top$$

as the angular velocity of the body-fixed frame with respect to the Earth-fixed frame expressed in the body-fixed frame (from now on: body-fixed angular velocity). The vector $\dot{\eta}_2$ does not have a physical interpretation and it is related to the body-fixed angular velocity by a proper Jacobian matrix

$$v_2 = J_{k,o}(\eta_2) \dot{\eta}_2. \quad (43.2)$$

Table 43.2 Common notation for the motion of a marine vehicle

		Forces and moments	v_1, v_2	η_1, η_2
Motion in the x-direction	Surge	X	u	x
Motion in the y-direction	Sway	Y	v	y
Motion in the z-direction	Heave	Z	w	z
Rotation about the x-axis	Roll	K	p	ϕ
Rotation about the y-axis	Pitch	M	q	θ
Rotation about the z-axis	Yaw	N	r	ψ

The matrix $J_{k,o} \in \mathbb{R}^{3 \times 3}$ can be expressed in terms of the Euler angles as

$$J_{k,o}(\eta_2) = \begin{pmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{pmatrix}, \quad (43.3)$$

where c_α and s_α are abbreviations for $\cos(\alpha)$ and $\sin(\alpha)$, respectively. The matrix $J_{k,o}(\eta_2)$ is not invertible for every value of η_2 . In detail, it is

$$J_{k,o}^{-1}(\eta_2) = \frac{1}{c_\theta} \begin{pmatrix} 1 & s_\phi s_\theta & c_\phi s_\theta \\ 0 & c_\phi c_\theta & -c_\theta s_\phi \\ 0 & s_\phi & c_\phi \end{pmatrix}, \quad (43.4)$$

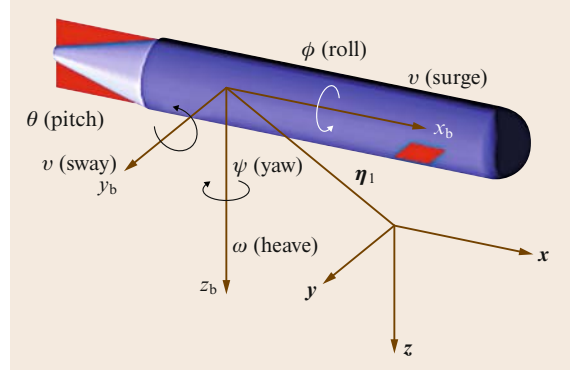
that it is singular for $\theta = (2l+1)\pi/2$ rad, with $l \in \mathbb{N}$, i. e., for a pitch angle of $\pm\pi/2$ rad.

The rotation matrix R_I^B , needed in (43.1) to transform the linear velocities, is expressed in terms of the Euler angles by

$$R_I^B(\eta_2) = \begin{pmatrix} c_\psi c_\theta & s_\psi c_\theta & -s_\theta \\ -s_\psi c_\phi + c_\psi s_\theta s_\phi & c_\psi c_\phi + s_\psi s_\theta s_\phi & s_\phi c_\theta \\ s_\psi s_\phi + c_\psi s_\theta c_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi & c_\phi c_\theta \end{pmatrix}. \quad (43.5)$$

Table 43.2 shows the common notation used for marine vehicles according to the Society of Naval Architects and Marine Engineers (SNAME) notation [43.6]; a sketch is shown in Fig. 43.2.

As for any representation of a rigid body's orientation several possibilities arise, among them, the use

**Fig. 43.2** Motion variables for an underwater vehicle

of a four-parameter description given by quaternions. The term *quaternion* was introduced by Hamilton in 1840, 70 years after the introduction of a four-parameter rigid-body attitude representation by Euler. An introduction to alternative orientation representations can be found in Chap. 1 and concerning the marine environment in [43.7].

It is useful to collect the kinematic equations in six-dimensional matrix forms. Let us define the vector $\eta \in \mathbb{R}^6$ as

$$\eta = \begin{pmatrix} \eta_1 \\ \eta_2 \end{pmatrix} \quad (43.6)$$

and the vector $v \in \mathbb{R}^6$ as

$$v = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}, \quad (43.7)$$

and, defining the matrix $J_e(R_B^I) \in \mathbb{R}^{6 \times 6}$

$$J_e(R_B^I) = \begin{pmatrix} R_I^B & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & J_{k,o} \end{pmatrix}, \quad (43.8)$$

where the rotation matrix R_I^B is given in (43.5) and $J_{k,o}$ is given in (43.3),

$$v = J_e(R_B^I) \dot{\eta}. \quad (43.9)$$

The inverse mapping, given the block-diagonal structure of J_e , is given by

$$\dot{\eta} = J_e^{-1}(R_B^I) v = \begin{pmatrix} R_I^B & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & J_{k,o}^{-1} \end{pmatrix} v, \quad (43.10)$$

where $J_{k,o}^{-1}$ is given in (43.4).

Defining as

$$\tau_v = \begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix},$$

the vector of generalized forces, where

$$\tau_1 = (X \ Y \ Z)^\top, \quad (43.11)$$

the resultant forces acting on the rigid body expressed in a body-fixed frame, and

$$\tau_2 = (K \ M \ N)^\top, \quad (43.12)$$

the corresponding resultant moment to the pole O_b , it is possible to rewrite the Newton–Euler equations of motion of a rigid body moving in the space as:

$$M_{RB}\dot{v} + C_{RB}(v)v = \tau_v. \quad (43.13)$$

The derivation of (43.13) can be found in Chap. 2.

The matrix M_{RB} is constant, symmetric, and positive definite, i. e., $\dot{M}_{RB} = 0$ and $M_{RB} = M_{RB}^\top > 0$. Its unique parametrization is of the form

$$M_{RB} = \begin{pmatrix} mI_3 & -mS(r_C^b) \\ mS(r_C^b) & I_{O_b} \end{pmatrix}, \quad (43.14)$$

where r_C^b is the 3×1 distance vector to the center of gravity (CG) expressed in the body-fixed frame, I_3 is the 3×3 identity matrix, and I_{O_b} is the inertia tensor expressed in the body-fixed frame $S(x)$ is the matrix operator performing the cross product between two (3×1) vectors

$$S(x) = \begin{pmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{pmatrix}.$$

On the other hand, there does not exist a unique parametrization of the matrix C_{RB} , which represents the Coriolis and centripetal terms. It can be demonstrated that the matrix C_{RB} can always be parameterized such that it is skew symmetric, i. e.,

$$C_{RB}(v) = -C_{RB}^\top(v) \quad \forall v \in \mathbb{R}^6; \quad (43.15)$$

explicit expressions for C_{RB} can be found in [43.7].

Notice that (43.13) can be greatly simplified if the origin of the body-fixed frame is chosen to be coincident with the central frame, i. e., $r_C^b = 0$.

Hydrodynamic Generalized Forces

Equation (43.13) represents the motion of a rigid body in an empty space, while dealing with ships or underwater vehicles requires the consideration of the presence of

the hydrodynamics generalized forces, i. e., the forces and moments caused by the presence of the fluid. In hydrodynamics it is common to assume that the generalized hydrodynamics forces on a rigid body can be linearly superimposed [43.8]; in particular, those are separated into radiation-induced forces, environmental disturbances, and restoring forces due to gravity and buoyancy.

Radiation-induced forces are defined as the *forces on the body when the body is forced to oscillate with the wave excitation frequency and there are no incident waves*; these can be identified as the sum of the *added mass* due to the inertia of the surrounding fluid and the *radiation-induced potential damping* due to the energy dissipated by generated surface waves.

Environmental disturbances can be identified as the generalized forces caused by the wind, the waves, and the ocean current.

The overall equations of motions can therefore be written in matrix form as [43.7, 9, 10]:

$$M_v\dot{v} + C_v(v)v + D_v(v)v + g_v(R_B^I) = \tau_v, \quad (43.16)$$

where $M_v = M_{RB} + M_A$ and $C_v = C_{RB} + C_A$ also include the added mass terms.

In the following subsections these generalized forces, specific to the marine environment, will be briefly discussed.

Added Mass and Inertia

When a rigid body is moving in a fluid, the additional inertia of the fluid surrounding the body that is accelerated by the movement of the body has to be considered. This effect can be neglected in industrial robotics since the density of the air is much lower than the density of a moving mechanical system. In underwater applications, however, the density of the water, $\rho \approx 1000 \text{ kg/m}^3$, is comparable with the density of the vehicles. In particular, at 0°C , the density of freshwater is 1002.68 kg/m^3 ; for sea water with 3.5% salinity it is $\rho = 1028.48 \text{ kg/m}^3$.

The fluid surrounding the body is accelerated with the body, so a force is necessary to achieve this acceleration, while the fluid exerts a reaction force which is equal in magnitude and opposite in direction. This reaction force is the added mass contribution. The added mass is not a quantity of fluid to add to the system such that it has an increased mass. Different properties hold with respect to the 6×6 inertia matrix of a rigid body due to the fact that the added mass is a function of the body's surface geometry.

The hydrodynamic force along \mathbf{x}_b due to the linear acceleration in the \mathbf{x}_b -direction is defined as:

$$X_A := -X_{\dot{u}}\dot{u}, \quad \text{where} \quad X_{\dot{u}} := \frac{\partial X}{\partial \dot{u}},$$

where the symbol ∂ denotes the partial derivative. In the same way it is possible to define all the remaining 35 elements that relate the six force/moment components $(X \ Y \ Z \ K \ M \ N)^\top$ to the six linear/angular accelerations $(\dot{u} \ \dot{v} \ \dot{w} \ \dot{p} \ \dot{q} \ \dot{r})^\top$. These elements can be grouped into the added mass matrix $\mathbf{M}_A \in \mathbb{R}^{6 \times 6}$. Usually, all the elements of the matrix are nonzero.

In general, added mass and potential damping will be frequency dependent and depend on the forward speed. This is also the case for certain viscous damping terms (skin friction, roll damping, etc.). This gives a pseudo differential equation describing the frequency response of the vehicle. Since some of the coefficients depend on the frequency this is not an ordinary differential equation (ODE). The frequency equation, however, can be transformed to the time domain using the concepts described in [43.11] and [43.12], and recently in [43.13]. The resulting equation is an ODE where the added inertia matrix \mathbf{M}_A is constant, speed independent, and positive definite:

$$\mathbf{M}_A = \mathbf{M}_A^\top > \mathbf{0}, \quad \dot{\mathbf{M}}_A = \mathbf{0}. \quad (43.17)$$

This result is well known from ship hydrodynamics; see [43.14] for instance. The matrix \mathbf{M}_A can be computed using numerical programs such as WAMIT or Matlab, based on the US Air Force Digital Datcom [43.15]; in this case, the infinity-frequency result should be used, that is, $\mathbf{M}_A = \mathbf{A}(\infty)$ where $\mathbf{A}(\omega)$ is the frequency-dependent added mass matrix. The potential damping matrix will be small compared to the viscous effects and drag/lift terms. Hence, this term can be set to zero for underwater vehicles. If the added mass is computed experimentally, it is common practice to symmetrize the results such that

$$\mathbf{M}_A = \frac{1}{2}(\mathbf{A}_{\text{exp}} + \mathbf{A}_{\text{exp}}^\top),$$

where \mathbf{A}_{exp} denotes the experimentally obtained added mass terms.

If the body is completely submerged in the water and is designed with port/starboard symmetry (xz -plane) as is common for underwater vehicles in six degrees-of-freedom (DOFs), the following structure of

the matrices \mathbf{M}_A can be considered:

$$\mathbf{M}_A = - \begin{pmatrix} X_{\dot{u}} & 0 & X_{\dot{w}} & 0 & X_{\dot{q}} & 0 \\ 0 & Y_{\dot{v}} & 0 & Y_{\dot{p}} & 0 & Y_{\dot{r}} \\ Z_{\dot{u}} & 0 & Z_{\dot{w}} & 0 & Z_{\dot{q}} & 0 \\ 0 & K_{\dot{v}} & 0 & K_{\dot{p}} & 0 & K_{\dot{r}} \\ M_{\dot{u}} & 0 & M_{\dot{w}} & 0 & M_{\dot{q}} & 0 \\ 0 & N_{\dot{v}} & 0 & N_{\dot{p}} & 0 & N_{\dot{r}} \end{pmatrix}. \quad (43.18)$$

The added mass coefficients can theoretically be derived by exploiting the geometry of the rigid body or numerically by strip theory [43.16].

In [43.17] the coefficients for the experimental AUV Phoenix of the Naval Postgraduate School (NPS) are reported. These coefficients have been derived experimentally, and the geometry gives a nondiagonal \mathbf{M}_A matrix. To provide an order of magnitude for the added mass terms, for the vehicle mass of about 5000 kg, the $X_{\dot{u}}$ is approximately -500 kg.

The added mass also makes an *added* Coriolis and centripetal contribution. It can be demonstrated that the matrix expression can always be parameterized such that

$$\mathbf{C}_A(\mathbf{v}) = -\mathbf{C}_A^\top(\mathbf{v}), \quad \forall \mathbf{v} \in \mathbb{R}^6,$$

whose symbolic expressions can be found in [43.3].

Hydrodynamic Damping

The hydrodynamic damping for marine vehicles is mainly caused by

- potential damping
- skin friction
- wave drift damping
- vortex shedding damping
- viscous damping

The radiation-induced potential damping due to forced body oscillations is commonly known as potential damping; its dynamic contribution is usually negligible with respect to, e.g., the viscous friction for underwater vehicles while it may be significant for surface vessels.

Linear skin friction is due to laminar boundary layers and can affect the low-frequency motion of the vehicle. Together with this effect, at high frequencies it is possible to observe a quadratic, or nonlinear, skin friction phenomenon caused by turbulent boundary layers.

Wave drift damping is the dominant dynamic damping effect in surge motion of surface vessels in high sea. It can be considered as an added resistance for boats advancing in waves; its drift is proportional to the square of the significant wave height. In the sway and yaw di-

Table 43.3 Lift and drag coefficient for a cylinder

Reynolds number	Regime motion	C_d	C_l
$R_n < 2 \times 10^5$	Subcritical flow	1	3–0.6
$2 \times 10^5 < R_n < 5 \times 10^5$	Critical flow	1–0.4	0.6
$5 \times 10^5 < R_n < 3 \times 10^6$	Transcritical flow	0.4	0.6

rections, however, its dynamic contribution is negligible with respect to the effect of vortex shedding.

A body moving in a fluid causes a separation of the flow; this can still be considered as laminar in the upstream while two antisymmetric vortices can be observed in the downstream. In case that the body is a cylinder moving in a direction normal to its axis, the result is a periodic force normal to both the velocity and the axis. This effect may cause the oscillation of cables and other underwater structures. However, concerning underwater vehicles, this effect is negligible for ROVs and may be counteracted by designing proper small control surfaces for torpedo-like AUVs.

Vortex shedding is an unsteady flow that takes place at special flow velocities (according to the size and shape of the cylindrical body). In this flow vortices are created at the back of the body, periodically from each side.

The viscosity of the fluid also causes dissipative forces. These are composed of drag and lift forces, the former being parallel to the relative velocity of the vehicle with respect to the water while the latter are normal to it. For a sphere moving in a fluid, the drag force can be modeled as [43.8]

$$F_{\text{drag}} = \frac{1}{2} \rho U^2 S C_d(R_n), \quad (43.19)$$

where ρ is the fluid density, U is the velocity of the sphere, S is the frontal area of the sphere, C_d is the nondimensional drag coefficient, and R_n is the Reynolds number. For a generic body, S is the projection of the frontal area along the flow direction. The drag force can be considered as the sum of two physical effects: the frictional contribution of the surface whose normal is perpendicular to the flow velocity, and the pressure contribution of the surface whose normal is parallel to the flow velocity. For a hydrofoil moving in a fluid, the lift force can be modeled as [43.8]

$$F_{\text{lift}} = \frac{1}{2} \rho U^2 S C_l(R_n, \alpha), \quad (43.20)$$

where S is now the area, C_l is the nondimensional lift coefficient, and α is the angle of attack, i.e., the angle between the relative velocity and the tangent to the surface. For small angles of attack, i.e., $|\alpha| < 10^\circ$, the

lift coefficient is approximatively proportional to α and rapidly decays to zero as α increases [43.18].

The drag and lift coefficients are therefore dependent on the Reynolds number, i.e., on the laminar/turbulent fluid motion

$$R_n = \frac{\rho |U| D}{\mu},$$

where D is the characteristic dimension of the body perpendicular to the direction of U and μ is the dynamic viscosity of the fluid. Table 43.3 reports the drag coefficients as a function of the Reynolds number for a cylinder [43.19].

A common simplification considers only linear and quadratic damping terms and group these into a matrix D_v as in (43.16) such that

$$D_v(\mathbf{v}) > \mathbf{0}, \quad \forall \mathbf{v} \in \mathbb{R}^6.$$

Gravity and Buoyancy

When a rigid body is completely or partially submerged in a fluid under the effect of the gravity two more forces have to be considered: the gravitational force and buoyancy. The latter is the only hydrostatic effect, i.e., it is not a function of the relative movement between the body and fluid.

Let us define as

$$\mathbf{g}^I = (0 \ 0 \ 9.81)^\top \text{m/s}^2$$

the acceleration of gravity. This effect is not constant but varies with depth, longitude, and latitude; however, this value is usually accurate enough for most applications except for inertial navigation systems.

For a completely submerged body the computation of these dynamic effects is straightforward. The submerged weight of the body is defined as $W = m \|\mathbf{g}^I\|$, while its buoyancy $B = \rho \nabla \|\mathbf{g}^I\|$, where ∇ is the volume of the body and m is its mass. The gravity force, which acts at the center of mass \mathbf{r}_C^B , is represented in the body-fixed frame by

$$\mathbf{f}_G(\mathbf{R}_I^B) = \mathbf{R}_I^B \begin{pmatrix} 0 \\ 0 \\ W \end{pmatrix},$$

while the buoyancy force, acting at the center of buoyancy \mathbf{r}_B^B , is represented in the body-fixed frame by

$$\mathbf{f}_B(\mathbf{R}_I^B) = -\mathbf{R}_I^B \begin{pmatrix} 0 \\ 0 \\ B \end{pmatrix}.$$

The 6×1 vector of force/moment due to gravity and buoyancy in the body-fixed frame, included in the left-hand side of the equations of motion, is represented by

$$\mathbf{g}_v(\mathbf{R}_I^B) = - \begin{pmatrix} \mathbf{f}_G(\mathbf{R}_I^B) + \mathbf{f}_B(\mathbf{R}_I^B) \\ \mathbf{r}_G^B \times \mathbf{f}_G(\mathbf{R}_I^B) + \mathbf{r}_B^B \times \mathbf{f}_B(\mathbf{R}_I^B) \end{pmatrix}.$$

In the following, the symbol $\mathbf{r}_G^B = (x_G \ y_G \ z_G)^\top$ (with $\mathbf{r}_G^B = \mathbf{r}_C^B$) will be used for the center of gravity. The expression for \mathbf{g}_v in terms of the Euler angles is

$$\mathbf{g}_v(\boldsymbol{\eta}_2) = \begin{pmatrix} (W - B)s_\theta \\ -(W - B)c_\theta s_\phi \\ -(W - B)c_\theta c_\phi \\ -(y_G W - y_B B)c_\theta c_\phi + (z_G W - z_B B)c_\theta s_\phi \\ (z_G W - z_B B)s_\theta + (x_G W - x_B B)c_\theta c_\phi \\ -(x_G W - x_B B)c_\theta s_\phi - (y_G W - y_B B)s_\theta \end{pmatrix}. \quad (43.21)$$

Current

Ocean currents are mainly caused by tidal movement, the atmospheric wind system over the sea surface, heat exchange at the sea surface, salinity changes, the Coriolis force due to the Earth's rotation, nonlinear waves, the major ocean circulations such as the Gulf Stream, the effect of setup phenomena or storm surges, and strong density gradients in the upper ocean. Currents can be very different due to local climatic and/or geographic characteristics; as an example, in fjords, the tidal effect can cause currents of up to 3 m/s, moreover, specific mathematical models exist for the various components [43.7].

Let us assume that the ocean current, expressed in the inertial frame, \mathbf{v}_c^I , is constant and irrotational, i. e.,

$$\mathbf{v}_c^I = (v_{c,x} \ v_{c,y} \ v_{c,z} \ 0 \ 0 \ 0)^\top$$

and $\dot{\mathbf{v}}_c^I = \mathbf{0}$; its effects can be added to the dynamics of a rigid body moving in a fluid simply by considering the *relative* velocity in the body-fixed frame

$$\mathbf{v}_r = \mathbf{v} - \mathbf{R}_I^B \mathbf{v}_c^I \quad (43.22)$$

in the derivation of the added Coriolis and centripetal and the damping terms.

$$\begin{aligned} \mathbf{M}_v \dot{\mathbf{v}} + \mathbf{C}_{RB}(\mathbf{v})\mathbf{v} + \mathbf{C}_A(\mathbf{v}_r)\mathbf{v}_r + \mathbf{D}_v(\mathbf{v}_r)\mathbf{v}_r + \mathbf{g}_v(\mathbf{R}_I^B) \\ = \boldsymbol{\tau}_v. \end{aligned} \quad (43.23)$$

Notice that the term $\mathbf{C}_A(\mathbf{v}_r)\mathbf{v}_r$ includes the important destabilizing effect known as the Munk moment [43.8].

If $\mathbf{D}_v(\mathbf{v}_r)$ is unknown, quadratic surge resistance and the cross-flow drag principle can be used to describe the dissipative forces and moments in surge, sway, and yaw [43.8]. Moreover:

$$\mathbf{C}_A(\mathbf{v}_r)\mathbf{v}_r + \mathbf{D}_v(\mathbf{v}_r)\mathbf{v}_r \approx (X_c \ Y_c \ 0 \ 0 \ 0 \ N_c)^\top; \quad (43.24)$$

for large relative current angles $|\beta_c - \psi|$, where β_c is the current direction, the cross-flow principles models the sway force Y_c and yaw moment N_c as

$$Y_c = \frac{\rho}{2} \int_L H(x) C_D(x) v_r^x(x) |v_r^x(x)| dx \quad (43.25)$$

$$N_c = \frac{\rho}{2} \int_L x H(x) C_D(x) v_r^x(x) |v_r^x(x)| dx, \quad (43.26)$$

where L is the vehicle length, $H(x)$ is the vehicle height, $C_D(x)$ is the two-dimensional drag coefficient, and $v_r^x(x) = v_r + rx$ is the relative cross-flow velocity at x . In practice $C_D(x)$ can be chosen as a constant between 0 and 1. The proper value can be determined by curve fitting of experimental data. Along the surge direction, however, the quadratic damping contribution X_c is still well represented by a term proportional to the square of the relative velocity, whose symbolic expression can be written as

$$X_c = -X_{u|u}|u_r|, \quad (43.27)$$

where $-X_{u|u} > 0$ is the quadratic surge damping coefficient, which can be found by curve fitting of experimental data or relating it to the drag coefficient C_d as in (43.19).

Alternatively, the computation of the quadratic surge resistance, nonlinear roll damping, and the cross-flow drag effect can be made by resorting to the Datcom database for aircraft, as shown in [43.20].

Model Properties

For completely submerged bodies in an ideal fluid moving at low velocity where there are no currents or waves, (43.16) satisfies the following properties:

- the inertia matrix is symmetric and positive definite, i. e.,

$$\mathbf{M}_v = \mathbf{M}_v^\top > \mathbf{0};$$

- the damping matrix is positive definite, i. e.,

$$\mathbf{D}_v(\mathbf{v}) > \mathbf{0};$$

- the matrix $C_v(\mathbf{v})$ is skew symmetric, i. e.,

$$C_v(\mathbf{v}) = -C_v^T(\mathbf{v}), \forall \mathbf{v} \in \mathbb{R}^6.$$

Hydrodynamic Modeling

The mathematical model of an underwater robot as expressed in (43.16) is of great importance; even when simplified it captures the most important part of the dynamics. Moreover, it is in a form appropriate for control design. A wide literature exists on AUV/ROV controllers whose stability relies on the properties reported above. On the other hand, there are working conditions in which these assumptions are no longer valid, i. e., when the AUV is traveling at high speed, or close to the surface, or when its shape does not allow geometric simplifications. The latter is the case of, e.g., several ROVs. In addition, it is still common to design the controllers for AUVs based on linearized models and to control ROVs with simple PID controllers.

These considerations justify a modeling effort to calculate the hydrodynamic terms more accurately with the aim of prediction, simulation, and performance analysis rather than control design. This can be done by switching from a coefficient-based approach, such as that presented above, to a component modeling method, the latter being based on computational fluid dynamics theory. In detail, each vehicle geometry, with its specific angle of attack and sideslip, is taken into consideration when computing the hydrodynamic forces/moments. This increased computational effort makes it possible to capture some dynamic effects, such as the vortex-induced roll moment, not justifiable with the coefficient-based approach.

Table 43.4 UUVs: possible instrumentation

Sensor	Measured variable
Inertial system	Linear acceleration and angular velocity
Pressure meter	Vehicle depth
Frontal sonar	Distance from obstacles
Vertical sonar	Distance from the bottom
Ground speed sonar	Relative velocity vehicle/bottom
Current meter	Relative velocity vehicle/current
Global positioning system	Absolute position at the surface
Compass	Orientation
Acoustic baseline	Absolute position in known area
Vision systems	Relative position/velocity
Acoustic Doppler current profiler	Water current at several positions

The control plant model is usually a simplified model that captures the most important parts of the dynamics. The most accurate model of the vehicle should be used for prediction and motion simulation.

43.2.2 Sensor Systems

Underwater vehicles are equipped with a sensor system devoted to enabling motion control as well as accomplishing the specific mission it has been commanded to complete. In the latter case, sensors developed for chemical/biological measurements or mapping may be installed, which is beyond the scope of this chapter.

AUVs need to operate underwater most of the time; one of the major problems with underwater robotics is in the localization task due to the absence of a single, proprioceptive sensor to measure the vehicle position. The global position system (GPS) cannot be used underwater. Redundant multisensor systems are commonly combined using state estimation or sensor fusion techniques to provide fault detection and tolerance capability to the vehicle. Table 43.4 lists the types of sensors and the corresponding variable measured commonly available for unmanned underwater vehicles (UUVs).

The sensors that can be found on an underwater vehicle are:

- Compass. A gyrocompass can provide an estimate of geodetic north accurate to a fraction of a degree. Magnetic compasses can provide estimates of magnetic north with an accuracy of less than 1° if carefully calibrated to compensate for magnetic disturbances from the vehicle itself. Tables or models can be used to convert from magnetic north to geodetic north.
- Inertial measurement unit (IMU). An IMU provides information about the vehicle's linear acceleration and angular velocity. These measurements are combined to form estimates of the vehicle's attitude including an estimate of geodetic (true) north from the most complex units. In most cases, for slow-moving underwater vehicles, an independent measurement of the vehicle's velocity is also required to produce accurate estimates of the translational velocity or relative displacement.
- Depth sensor. Measuring the water pressure gives the vehicle's depth. At depths beyond a few hundred meters, the equation of state of seawater must be invoked to produce an accurate depth estimate based on the ambient pressure [43.21]. With a high-quality

Table 43.5 JHUROV instrumentation

Measured variable	Sensor	Precision	Update rate
3-DOF vehicle position	SHARP acoustic transponder	0.5 cm	10 Hz
Depth	Foxboro/ICT model n. 15	2.5 cm	20 Hz
Heading	Litton LN200 IMU Gyro	0.01°	20 Hz
Roll and pitch	KVH ADGC	0.1°	10 Hz
Heading	KVH ADGC	1°	10 Hz

Table 43.6 ODIN III sensors summary

Measured variable	Sensor	Update rate
x-y vehicle position	8 sonars	3 Hz
Depth	Pressure sensor	30 Hz
Roll, pitch, and yaw	IMU	30 Hz

- sensor, these estimates are reliable and accurate, giving a small error of order 0.01%.
- Altitude and forward-looking sonar. These are used to detect the presence of obstacles and distance from the seafloor.
 - Doppler velocity log (DVL). By processing reflected acoustic energy from the seafloor and the water column from three or more beams, estimates of vehicle velocity relative to the seafloor and relative water motion can be obtained. Bottom-tracking velocity estimates can be accurate to ≈ 1 mm/s.
 - Global positioning system (GPS). This is used to localize the vehicle while on the surface to initialize or reduce drift of estimates from an IMU/DVL combination. GPS only works at the surface.
 - Acoustic positioning. A variety of schemes exist for determining vehicle position using acoustics. Long-baseline navigation can determine the position of the vehicle relative to a set of acoustic beacons anchored to the seafloor or on the surface through range estimates obtained from acoustic travel times. Ultrashort-baseline navigation uses phase information to determine direction from a cluster of hydrophones; this is most often used to determine the direction of the vehicle (in two dimensions) from a surface support vessel, which is then combined with an acoustic travel-time measurement to produce an estimate of relative vehicle position in spherical coordinates. These techniques will be discussed later in the localization section.

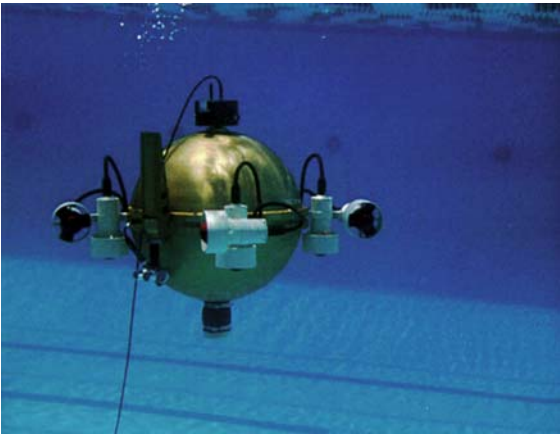


Fig. 43.3 The fully actuated AUV ODIN (courtesy of the Autonomous Systems Laboratory, University of Hawaii, <http://www.eng.hawaii.edu/~asl/>)

- Vision systems. Cameras can be used to obtain estimates of relative, and in some cases absolute, motion using a type of simultaneous localization and mapping (SLAM) algorithm [43.22] and used to perform tasks such as visual tracking of pipelines, station keeping, visual servoing or image mosaicking.
- As an example, Table 43.5 reports some data from the instrumentation of the ROV developed at the John Hopkins University [43.23], and Table 43.6 some data from the AUV ODIN III [43.24]. Reference [43.25] shows some data fusion results with a redundant sensorial system mounted on the AUV Oberon, while reference [43.26] reviews advances in navigation technology.

43.2.3 Actuating Systems

Marine vehicle are generally propelled by means of thrusters or hydrojets. In the case of ROVs with structural pitch–roll stability, there are usually four thrusters that provide holonomic mobility to the four remained DOFs, in particular, the depth is often decoupled and the vehicle is controlled on a plane in the surge, sway, and yaw DOFs. Those vehicles, being underactuated, cannot easily be used for interactive control by means of a manipulator due to the impossibility of counteracting the generalized forces exchanged with the manipulator’s base; in such case, six or more thrusters are required. AUVs generally have a torpedo-like shape and are used for mapping/exploration. They are propelled using one or two thrusters parallel to the fore–aft direction and

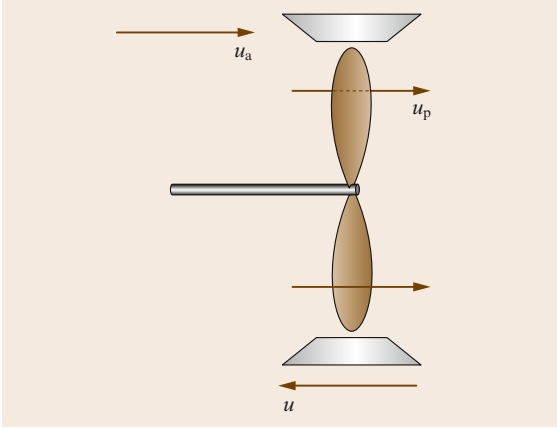


Fig. 43.4 Ambient water and axial flow velocities affecting thruster behavior

a fin and a rudder; this kind of propulsion is obviously nonholonomic and experiences a loss of mobility at low velocities. Hydrojets, also known as pump jets or water jets, are systems that create a jet of water for propulsion; they have certain advantages over thrusters such as a higher power density and usability in shallow water, but can provide thrust in one direction only.

Several efforts have been made to accurately and efficiently describe the mathematical model of a thruster; [43.28] reports a *one-state* model where the state is n , the propeller shaft speed. In [43.29] a *two-state* model is proposed to take into account the experimentally observed overshoot in the thrust; together with n , the additional state variable is u_p , the axial flow velocity in the propeller disc. In [43.30] a thruster model incorporating the effects of rotational fluid velocity and inertia on thruster responses is presented together with a method for experimentally determining nonsinusoidal lift/drag curves. A *three-state* model is described in [43.31]:

$$\begin{aligned} J_m \dot{n} + K_n n &= \tau - Q, \\ m_f \dot{u}_p + d_{f0} u_p + d_f |u_p| (u_p - u_a) &= T, \\ (m - X_{\dot{u}}) \dot{u} - X_{uu} u - X_{u|u}|u| &= (1-t)T, \end{aligned}$$

where J_m is the moment of inertia for the dc-motor/propeller, K_n is the linear motor damping coefficient, τ is the motor control input, Q is the propeller torque, m_f is the mass of water in the propeller control volume, u_p is the axial flow velocity in the propeller disc, d_{f0} and d_f are the linear and quadratic damping coefficients for control volume, respectively, u_a is the ambient water velocity, T is the propeller thrust, and t is the thrust deduction number (Fig. 43.4). In the

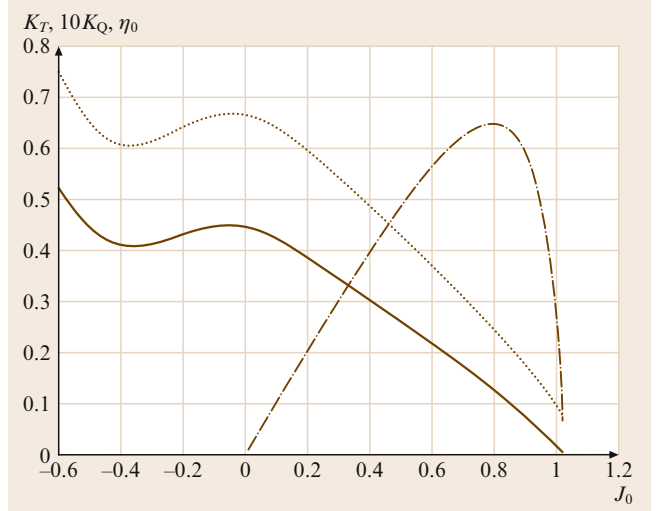


Fig. 43.5 Values of K_T (solid), $10K_Q$ (dotted), and η_0 (dash-dotted) as a function of J_0 [43.27]

case of steady-state motion, i. e., $\dot{u} = 0$, the ambient water velocity u_a is related to the surge by the *wake fraction number* w as:

$$u_a = (1 - w)u. \quad (43.28)$$

Notice also that the unmeasured variable u_p can be estimated using a nonlinear observer [43.31].

The *outputs* of the nonlinear three-state dynamic systems are the thrust T and the torque Q , which are functions of several variables; in the following, unsteady flow effects such as air suction, cavitation, the in-and-out-of-water (Wagner), boundary layer, and gust (Kuessner) effects will be neglected. This leads to a quasi-steady representation of the model:

$$T = \rho D^4 K_T(J_0) n |n|, \quad (43.29)$$

$$Q = \rho D^5 K_Q(J_0) n |n|, \quad (43.30)$$

where D is the propeller diameter and $K_T(J_0)$ and $K_Q(J_0)$ are the thrust and torque coefficients, respectively. The latter are function of the advance ratio J_0

$$J_0 = \frac{u_a}{nD}. \quad (43.31)$$

The open-water propeller efficiency in undisturbed water is given as the ratio of the work done by the propeller in producing a thrust force divided by the work required to overcome the shaft torque:

$$\eta_0 = \frac{u_a T}{2\pi n Q} = \frac{J_0}{2\pi} \cdot \frac{K_T}{K_Q}. \quad (43.32)$$

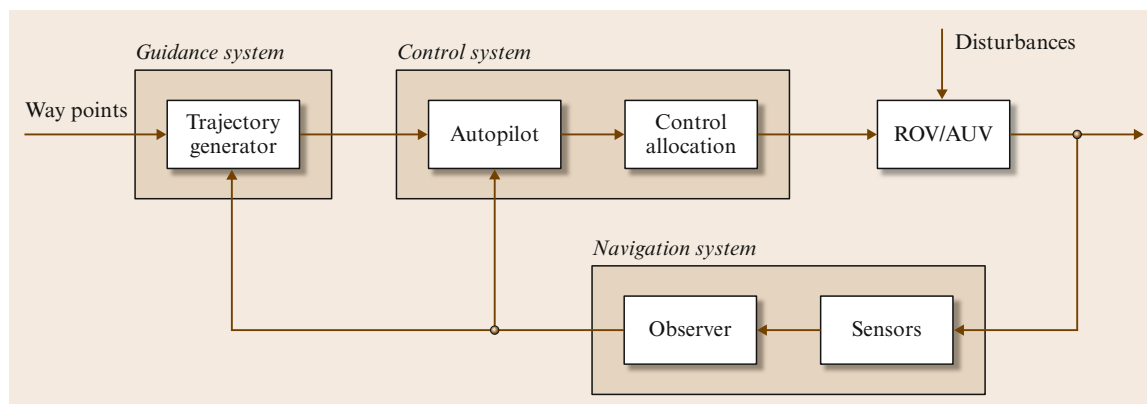


Fig. 43.6 Guidance, navigation, and control for an autonomous marine vehicle

Figure 43.5 shows the values of K_T , K_Q , and η_0 as functions of the advance ratio for the Wageningen B4-70 propeller [43.27].

Controlling a marine vehicle usually requires that desired forces/moments act on the vehicle's body; these generalized forces are mapped into desired thrusts to be provided by the propellers. There is, thus, a nontrivial control problem in that the motors are required to provide the appropriate propeller shaft speed n that satisfies the nonlinear relationship with the thrust T presented above.

To enable robustness with respect to possible failures, the actuating system is often redundant. In this case, the problem of allocation of the desired force/moment acting on the vehicle among the thrusters must also be solved. Reference [43.32] reports a survey of control allocation methods of ships and underwater vehicles.

43.2.4 Mission Control System

The mission control system (MCS) can be considered as the highest-level process running during an AUV's mission; it is responsible for achieving several control objectives. At the highest level it works as an interface between the operator, accepting his instructions in a higher-level language and decomposing those instructions into mission tasks according to the implemented software architecture. The mission tasks are generally concurrent and their handling depends on the vehicle state and environmental conditions; it is therefore the MCS that handles the tasks, eventually suppressing, sequencing, modifying, and prioritizing them. An MCS is also usually equipped with a graphical user interface (GUI) to report the mission state to the operator.

As for most advanced robotics applications, an efficient MCS should allow the use of complex robotic systems by users that do not necessarily know all of their technical details. An overview relevant to underwater mission control is given in [43.33], which includes an interesting classification of the MSCs in use in several laboratories according to which four major AUV control architectures were identified: the hierarchical, heterarchical, subsumption, and hybrid.

From a mathematical point of view, the MCS generally needs to be designed in order to be able to address hybrid dynamical systems, i.e., handling both event-driven and time-driven processes. In [43.34], e.g., the MSC developed at the Portuguese Instituto Superior Técnico (IST), named CORAL, is implemented by resorting to a Petri-net-based architecture that properly handles all the necessary tasks in order to manage navigation, guidance and control, sensing, communications, etc.

The motion-oriented operating system (MOOS), designed at the Massachusetts Institute of Technology, is a software tool capable of executing and coordinating a multitude of subsea operations. The MSC developed at the Naval Postgraduate School is in the framework of the behavioral control organized in three layers [43.35]; it is based on PROLOG, an artificial intelligence language for predicate logic.

43.2.5 Guidance and Control

The terms guidance and control can be defined as [43.7]:

Guidance is the action of determining the course, attitude, and speed of the vehicle, relative to

some reference frame (usually the Earth), to be followed by the vehicle,

Control is the development and application to a vehicle of appropriate forces and moments for operating point control, tracking, and stabilization. This involves designing the feedforward and feedback control laws.

Figure 43.6 shows the corresponding block diagram, in which the navigation component is also outlined.

Guidance of Underwater Vehicles

Guidance algorithms may benefit from a wide range of inputs, overall mission information, real-time operator input, environmental measured data such as the ocean current, environmental topological information such as a bathymetric map, exteroceptive sensors for obstacle avoidance, and obviously the vehicle state as output from the navigation system.

The vehicle may be required to follow a path, i.e., a curve geometrically represented in two or three dimensions, or a trajectory, i.e., a path with a specific time law assigned. Moreover, when the desired position is constant, the problem is called set-point regulation or maneuvering. The guidance problem is commonly decomposed into simple subtasks of lower dimension: an attitude control problem and a path control. Moreover, attitude is usually considered as a simple depth set-point with null roll and pitch and the path is usually a line in the horizontal plane.

One of the most common guidance approaches is based on the generations of way-points. Those are usually stored in a database and are used to generate the vehicle path/trajectory; a *passing* velocity, in fact, may be defined together with the Cartesian coordinates of the points. The simplest way to connect the way-points is to use the segments connecting two successive way-points. Efficient way-point-based guidance approaches need to take into account the presence of the current and the eventual nonholonomicity of the vehicle [43.36]. A technique for adaptively tracking bathymetric contours by proper generation of way-points is presented in [43.37]; environment information is acquired by mean of a single vertical sonar. An alternative method is based on *line-of-sight* guidance [43.38–40]. In this case, the heading control is computed by considering as input the angle formed by the vector from the vehicle to the next way-point rather than requiring the vehicle to exactly follow the line segment between the current and the following way-point. Special care needs to be paid to

the docking maneuver with algorithms designed on the scope [43.41].

By combining vision-based guidance with a neuro-controller trained by reinforcement learning, in [43.42], an algorithm aimed at a hold station on a reef or swimming along a pipe has been presented. In [43.43] guidance for AUVs specifically involved in a pre-deployment survey of the sea bottom and visual inspection of pipelines is given. Reference [43.44] reports a specific guidance system aimed at mine avoidance for AUVs. Based on a three-dimensional discretization of the environment, the path-planning technique consists of computing a safe path avoiding the unsafe cells of the map. Due to the poor manoeuvrability at low speed under some conditions, the vehicle has to make a 360° turn to avoid stopping and to map the environment close to it before generating a safe path.

A deep discussion on guidance for surface and underwater vehicles can be found in [43.3, 7].

Control of Underwater Vehicles

Control of underwater vehicles needs to consider the different operating conditions and actuating configurations in which a submerged vehicle is required to operate. In particular, there are three main control problems.

- An AUV traveling at high speed (> 1 m/s) generally equipped with at least one thruster aligned in the fore–aft direction and at least two control surfaces (stern and rudder).
- An underactuated ROV, with high metacentric stability, i.e., structurally stable in roll and pitch, and equipped with at least four thrusters.
- A fully actuated AUV equipped with at least six thrusters.

AUVs equipped with control surfaces are under-actuated vehicles mainly used for survey/exploration missions. Inheriting the common practice of submarine control, they are not allowed to perform arbitrary motions in six DOFs but are rather designed to perform specific movements such as: cruising along a given direction at constant depth, steering at constant depth, or diving. Marine experience and mathematical insight, in fact, demonstrate that these movements are lightly coupled in dynamic terms. For these vehicles, moreover, specific manoeuvres such as homing or docking require special capabilities [43.41]. This requires the design of vehicles that are structurally stable in the roll DOF. Cruising requires control of the surge velocity $u(t)$; steering requires control of the sway velocity $v(t)$ and the yaw DOF $r(t)$, $\psi(t)$, diving requires control of the

heave DOF $\omega(t)$, $z(t)$ and the pitch DOF $q(t)$, $\theta(t)$. The simplest configuration of actuators that can control an AUV through these movements is composed of one thruster aligned along the fore–aft direction, one stern, and one rudder; the control variables, thus, are the propeller speed and the deflection of the fins. Several approaches can then be considered to solve this control problem, among them, in [43.45] the sliding mode control is proposed, while [43.46] presents an adaptive sliding mode control for the diving manoeuvre. Reference [43.17] reports a successful implementation of multivariable sliding mode control on the NPS AUV II, later also implemented on the NPS ARIES AUV [43.47]. As the model of an AUV traveling at high speed is nonlinear and coupled, the tuning of the parameters is mainly based on a linearized model around the working conditions.

From a descriptive point of view, an ROV is mainly a box-shaped underwater vehicle equipped with tools such as a video camera or robot manipulator, while its payload is often variable depending on the task. It is remotely operated and physically connected to another vehicle, either an underwater or a surface vessel. It is mainly designed to travel at low speed and it is structurally stable in roll and pitch, while its depth, surge, sway, and yaw are independently controllable. Due to the absence of a specific shape, the varying payload, and the relatively low required performances, it is common to control a ROV by means of single-input single-output (SISO) controllers. Moreover, the PID approach is often used due to its simplicity. A two-layer guidance and control architecture for the ROV Romeo is given in [43.48].

Control of a fully actuated AUV in six DOFs is needed in the case of, e.g., an interaction task performed by a manipulator mounted on a vehicle; the latter, in fact, needs to provide all the force/moment components in order to counteract the presence of the manipulator dynamically. This problem is kinematically similar to the problem of controlling a satellite in six DOFs; the underwater environment, however, makes it significantly different from the dynamic point of view. In kinematic terms the main issue is in implementing a suitable policy for orientation control; any three-parameter representation of orientation, in fact, experiences representation singularities (Chap. 1). This problem may be overcome by resorting to a redundant representation of the orientation such as the quaternion. Most of the six-DOF controllers proposed in the literature are based on (43.16), which model the simplified effect of the hydrodynamic terms and which have very

similar properties as the equations of motion of an industrial manipulator. Based on this, it is obviously possible to find a collection of approaches inherited from classical robotics, see, e.g., [43.3, 7] for some examples. In [43.49], some specific considerations for the underwater environment lead to a quaternion-based, adaptive controller; it is worth noticing that adaptive control requires a suitable, and simplified, expression for the hydrodynamic terms. In [43.50] a comparison among several 6-DOF controllers is made.

43.2.6 Localization

Localization in the underwater environment can be a complex task, mainly due to the absence of a single external sensor that gives the vehicle position such as, e.g., the GPS for outdoor ground vehicles; moreover, the environment is often poorly structured.

One of the most reliable methods is based on the use of acoustic systems such as the baseline systems: the long-baseline system (LBL), the short-baseline system (SBL), and the ultrashort-baseline system (USBL). These systems are based on the presence of a transceiver mounted on the vehicle and a variable number of transponders located in known positions. The transceiver's distance from each transponder can be measured via the measurement of an echo delay; from this information the position of the vehicle can be calculated by basic triangulation operations. The USBL can be used with a single transponder, which is usually mounted on a surface ship whose position is measured by GPS.

Another localization system is called terrain-aided navigation and is based on the use of terrain elevation maps; bathymetric maps are available, especially in the case of well-known locations such as harbors where they usually have a resolution of ≈ 1 m. In this case, the vehicle position is obtained by filtering the information coming from a downward-looking sonar. In [43.51], a particle filter approach was used to localize an AUV in Sydney harbor.

Moving vehicles may be equipped with an IMU or DVL in order to measure its velocity and/or acceleration. This data can then be integrated to estimate the vehicle position. This kind of information is subject to the drift phenomenon and may not be reliable for long-duration runs or may become cost ineffective if accurate IMU devices are needed.

Relative localization can be obtained by resorting to any device that provides information about the relative position of the vehicle with respect to the environment,

even in the absence of a map. In this case, by filtering the distance measurements taken along the motion, the vehicle's position can be measured. This is the case, e.g., of sonar or vision-based localization techniques [43.52].

Often, the techniques presented above are used together in a redundant system and the effective position is obtained by resorting to sensor fusion techniques such as the Kalman filtering approach.

Simultaneous Localization and Mapping

Simultaneous localization and mapping (SLAM), also known as concurrent mapping and localization (CML), is a wide topic in mobile robotics. The problem can be formulated as the requirement for a mobile robot to be placed in an unknown environment and progressively build a map while locating itself inside the map. Chapter 37 discusses this topic in detail. For the marine environment an additional issue arises due by the large-scale map that needs to be used for long-duration missions; [43.53] implements a decoupled stochastic mapping to handle this computational problem in an extended Kalman filter. Terrain-aided navigation with the use of a scanning sonar is implemented in [43.54]. Reference [43.55] uses long-baseline range measurements as the input for a nonlinear least-squares approach solved by the Gauss–Newton method; both the initially unknown position of the transponders and the vehicle position are estimated. An interesting survey on navigation and SLAM for underwater vehicles is given in [43.26].

43.2.7 Underwater Manipulation

A manipulator may be mounted on an AUV or a ROV in order to accomplish interaction operations. In this

case, the vehicle needs to be fully actuated to counteract the forces and moments generated by the manipulator's base. By considering a manipulator with n links, thus six DOFs, the underwater vehicle manipulator system (UVMS) is a $(6+n)$ -DOF robotic system whose velocity vector is

$$\zeta = (v_1^\top \ v_2^\top \ \dot{q}^\top)^\top, \quad (43.33)$$

where $q \in \mathbb{R}^n$ is the vector collecting the manipulator joints positions.

Repeating the same considerations as for an underwater vehicle, it is possible to write the equations of motions of an UVMS in matrix form as

$$M(q)\dot{\zeta} + C(q, \zeta)\zeta + D(q, \zeta)\zeta + g(q, R_B^I) = \tau, \quad (43.34)$$

where $M \in \mathbb{R}^{(6+n) \times (6+n)}$ is the inertia matrix, including added mass terms, $C(q, \zeta) \in \mathbb{R}^{6+n}$ is the vector of the Coriolis and centripetal terms, $D(q, \zeta) \in \mathbb{R}^{6+n}$ is the vector of dissipative effects, and $g(q, R_B^I) \in \mathbb{R}^{6+n}$ is the vector of gravity and buoyancy effects. The relationship between the generalized forces τ and the control input is given by:

$$\tau = \begin{pmatrix} \tau_v \\ \tau_q \end{pmatrix} = \begin{pmatrix} B_v & 0_{6 \times n} \\ 0_{n \times 6} & I_n \end{pmatrix} u = Bu, \quad (43.35)$$

where $u \in \mathbb{R}^{p_v+n}$ is the vector of the control input. Notice that, while for the vehicle a generic number $p_v \geq 6$ of control inputs is assumed, for the manipulator it is supposed that n joint motors are available.

Under this hypothesis, which can be considered as reasonable at low velocity, it holds that:

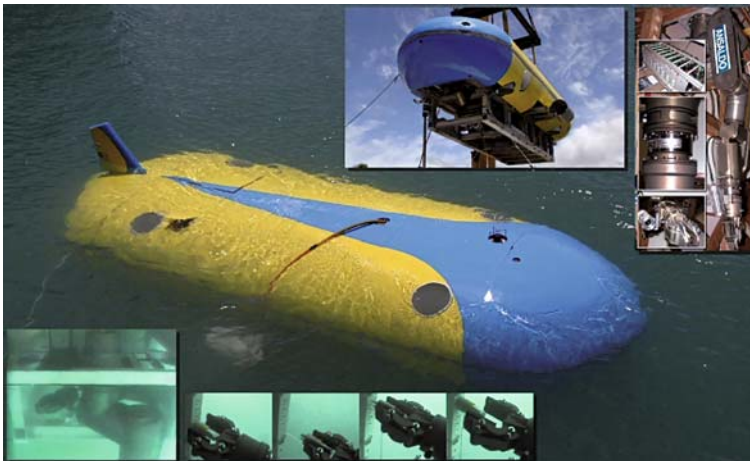


Fig. 43.7 An underwater vehicle–manipulator system: SAUVIM (courtesy of Autonomous Systems Laboratory, University of Hawaii, <http://www.eng.hawaii.edu/~asl/>)

- the inertia matrix \mathbf{M} of the system is symmetric and positive definite;
- for a suitable choice of the parametrization of \mathbf{C} and if all the single bodies of the system are symmetric, $\dot{\mathbf{M}} - 2\mathbf{C}$ is skew symmetric;
- the matrix \mathbf{D} is positive definite.

In [43.19] the mathematical model written with respect to the Earth-fixed-frame vehicle position and the manipulator end-effector can be found. However, it must be noted that, in this case, a six-dimensional manipulator is considered in order to have a square Jacobian to work with; moreover, kinematic singularities need to be avoided.

The equations of motion of UVMSs in matrix form presented in (43.34) are formally similar to the equations of motion of ground fixed manipulators (Chap. 2) for which a wide control literature exists. This has suggested a suitable translation/implementation of existing control algorithms. However, some differences, crucial from the control aspect, need to be underlined. UVMSs are complex systems characterized by several strong constraints:

- uncertainty in the model knowledge, mainly due to the poor knowledge about the hydrodynamic effects
- the complexity of the mathematical model
- the kinematic redundancy of the system
- the difficulty in controlling the vehicle in hovering, mainly due to poor thruster performance
- the dynamic coupling between the vehicle and the manipulator
- the low bandwidth of the sensor readings

In 1996 *McLain* et al. [43.56] presented a control law for UVMSs with some interesting experimental results conducted at the Monterey Bay Aquarium Research Institute (MBARI). A one-link manipulator was mounted on the OTTER vehicle controlled in all six DOFs by means of eight thrusters. A coordinated control system was then implemented to improve the tracking errors of the end effector.

The monograph [43.50] is focused on the modeling and control issues of such systems, and can be considered as a reference for further reading. Moreover, interaction with the environment is also discussed.

Currently, remotely operated manipulator are standard equipment for several underwater ROVs. However, autonomous manipulation is still a research challenge. Figure 43.7 shows the SAUVIM vehicle, one of the first semiautonomous underwater vehicle manipulator systems, developed at the Autonomous Systems Labo-

ratory, University of Hawaii. A similar research project, ALIVE, was funded by the Fifth Framework Program of the European Community [43.2].

43.2.8 Fault Detection/Tolerance

Generally, AUVs must operate over long periods of time in unstructured environments in which an undetected failure could cause the loss of the vehicle. Failure detection and a fault-tolerant strategy are required to determine whether a mission must be terminated in the safest manner possible or if the vehicle can continue in a diminished capacity. An example is the case of the arctic mission of Theseus [43.57].

In the case of the use of ROVs, a skilled human operator is in charge of commanding the vehicle; a failure detection strategy is then of help in the human decision-making process. Based on the information detected, the operator can decide on vehicle rescue or to terminate the mission by, e.g., turning off a thruster.

Fault detection is the process of monitoring a system in order to recognize the presence of a failure; fault isolation or diagnosis is the capability to determine which specific subsystem is subject to failure. Often in the literature there is a certain overlap in the use of these terms. Fault tolerance is the capability to complete the mission in the case of the failure of one or more subsystems; it is also known as fault control, fault accommodation, or control reconfiguration. In the following the terms fault detection/tolerance will be used.

The characteristics of a fault detection scheme are the capability to isolate the detected failure, sensitivity in terms of the magnitude of the failure that can be detected, and robustness in the sense of the capability to continue working properly in non-nominal conditions. The requirements of a fault-tolerant scheme are reliability, maintainability, and survival. The common concept is that, to overcome the loss of capability due to a failure, a kind of redundancy is required in the system.

In this section, a survey of existing fault-detection and fault-tolerant schemes for underwater vehicles is presented. For these specific systems, if proper strategies are applied, a hardware/software (HW/SW) sensor or thruster failure can be successfully handled. In some conditions, the fault-detection scheme must also be able to diagnose some external abnormal working conditions such as a multipath phenomena affecting the echo-sounder system. It is worth noticing that, for autonomous systems such as AUVs, space systems or aircraft, a fault-tolerant strategy is necessary to safely recover the damaged vehicle and, obviously, there is no

panic button in the sense that the choice of turning off the power or activating some kind of brake is not available.

Most fault-detection schemes are model based [43.58, 59] and consider the dynamic relationship between the actuators and vehicle behavior or the specific input–output thruster dynamics. In general, fault-detection/tolerance theory has been applied to the specific case of the underwater environment even if only a few papers report experimental results; see [43.60] for a survey on this topic.

Most fault-tolerant schemes consider a thruster-redundant vehicle that, after a fault has occurred in one of its thrusters, is still actuated in six DOFs. Based on this assumption a reallocation of the desired forces on the vehicle over the working thrusters is performed [43.61]. Of interest is also the study of reconfiguration strategies if the vehicle becomes underactuated.

Possible Failures

Underwater vehicles are currently equipped with several sensors in order to provide information about their localization and velocity. The problem is not easy. No single, reliable sensor is available that gives the required position/velocity measurement, or information about the environment such as the presence of obstacles. For this reason the use of sensor fusion by, e.g., a Kalman filtering approach, is a common technique to provide the controller with the required variables. This structural redundancy can be used to provide fault-detection capabilities to the system.

For each of the sensors listed in Sect. 43.2.2 failure can consist of an output of zero if, e.g., there is an electrical trouble, or a loss of meaning. It can be considered as sensor failure also an external disturbance such as a multipath reading of the sonar that can be interpreted as a sensor fault and correspondingly detected.

Thruster blocking occurs when a solid body is present between the propeller blades. It can be checked by monitoring the current required by the thruster. This was observed, e.g., during the Antarctic mission of Romeo [43.62], in that case caused by a block of ice. During the same mission a thruster also flooded with water. The consequence was an electrical dispersion, causing an increasing blade rotation velocity and thus a thruster force higher than desired.

A possible consequence of different failures of the thrusters is the zeroing of the blade rotation. The thruster in question thus simply stops working. This has been intentionally experienced during experiments with, e.g., ODIN [43.59, 61], Roby 2 [43.58], and Romeo [43.62].

Other failures include a hardware/software crash or the occurrence of fin sticking or loss. A very common type of failure involves the loss of electrical isolation due to seawater intrusion into underwater electrical cables or connectors. Such a condition can be detected through a technique called ground-fault monitoring. Should this occur, electrical power must be removed from the affected device.

43.2.9 Multiple Underwater Vehicles

A growing research effort has recently been devoted to developing strategies to design coordinated control for underwater vehicles. The use of multiple AUVs, in fact, might improve overall mission performance as well as provide greater tolerance to failures (Chap. 40). Specific applications of this method in the underwater environment might include the naval mine counter-measure problem, harbor monitoring, and inspection, exploration, and mapping of large areas. AUVs might be coordinated with one or more surface vessels or connected to ground or aerial vehicles to form a coordinated network of heterogeneous autonomous robots.

Beside several institutions that have developed simulation packages for multiple-AUV operations, the use of real multiple AUVs is being considered for the adaptive sampling and forecasting plan of the Autonomous Ocean Sampling Network, formed by several research institutions such as (for the robotic components) Caltech, MBARI, Princeton, and WHOI [43.63]. Adaptive sampling is also being investigated at the Autonomous Systems and Controls Laboratory of Virginia Tech, which has developed five AUVs [43.64]. The Australian National University is currently working on a shoal of small, autonomous robots, named Serafina [43.65]. At Instituto Superior Técnico (IST), work is ongoing on the coordination between an AUV and a catamaran [43.66], i. e., a multirobot system constituted by heterogeneous autonomous vehicles.

43.3 Applications

Underwater robots currently play prominent roles in a number of scientific, commercial, and military tasks.

Remotely teleoperated vehicles are very well established in all these areas, and are becoming increasingly auto-

mated to relieve the burden on human operators and to improve performance. Increasingly, autonomous underwater vehicles are finding application in these areas as well. Presently, AUVs are used almost exclusively for survey work, but sampling and other intervention tasks are becoming more feasible. Additionally, the line between ROVs and AUVs continues to blur, as systems that have the best properties of both evolve.

The offshore oil and gas industry relies heavily on ROVs for installation, inspection, and servicing of platforms, pipelines, and subsea production facilities. As the search for oil and gas goes deeper, this trend can only continue. The Marine Technology Society estimates that there are over 435 *work-class* ROVs operating in the commercial offshore industry today. AUVs are now beginning to appear in the commercial offshore industry for survey tasks, and concepts for hybrid systems that can perform intervention tasks are now appearing. The goal is not only for these robotic vehicles to replace human divers or human-occupied vehicles, but to enable an entire new generation of subsea equipment that is serviced without intervention by drill ships or other heavy-lifting vessels. This holds the prospect of greatly reduced cost.

Scientific demand for ROVs and AUVs is also increasing dramatically. Scientific applications for ROVs include survey, inspection, and sampling tasks previously performed by human-occupied submersibles or towed vehicles. While ROVs operating for science are not nearly as numerous as those in the offshore oil and gas industry, they are becoming commonplace. Most nations involved in global seafloor studies have several vehicles. Like the vehicles for the commercial offshore sector, these vehicles are becoming increasingly automated. High-quality electronic imaging, including high-definition television, is becoming increasingly common. Scientific ROVs are now equipped with sophisticated sampling devices for sampling animals, microbes, caustic hydrothermal vent fluids, and a variety of rock samples. Moreover, ROVs are also used to deploy and operate seafloor experiments, which can involve difficult tasks such as drilling and delicate placement of instruments.

ROVs have also emerged as powerful tools for investigating underwater shipwrecks and other cultural sites. Applications include forensic investigations of modern shipwrecks to determine the cause of sinking, archaeology, and salvage. For archaeology, the goals are the same as for excavation on land: detailed mapping fol-

lowed by careful excavation. Beyond diver depths, ROVs are the preferred method for these investigations. Great progress has been made in the detailed mapping phase, and capabilities for excavation are evolving. Unfortunately, the same technology also opens the possibility for shipwrecks to be looted for financial gain, which usually results in the loss of the most valuable historical information.

After a long period of skepticism, AUVs are now accepted for scientific tasks. Presently, AUVs most often perform mapping tasks while tended by a vessel. Specific mapping tasks include seafloor bathymetry, sidescan sonar imaging, magnetic field mapping, hydrothermal vent localization, and photo surveys. AUVs have been shown to improve productivity and data quality compared to towed and tethered systems. They have also operated in environments where no other means of gathering data is possible, such as under ice shelves. Likewise, the increasing availability of sophisticated in situ chemical sensors, biological sensors, and mass spectrometers now allows AUVs to build spatial and temporal maps of environmental features that could previously only be studied by bringing samples back to the laboratory. Plans are now underway for AUVs that can dock to subsea moorings or observatory nodes to recharge batteries and receive new instructions.

The military has always been a leader in the development of underwater robotic capabilities. They pioneered ROVs for tasks such as recovering test weapons and deep-sea salvage, and present-day commercial and scientific ROVs have descended directly from these early systems. Likewise, military interests are presently pushing AUV technology very hard. Many different countries operate AUVs for military surveys, gathering environmental data as well as searching for hazards such as mines. An operational success was achieved in surveying for mines in the Persian Gulf harbor of Umm Qasr using REMUS vehicles. AUVs in development will not only be able to detect mines, but to disable them. Bolder, more innovative concepts are also in development. These include networks of AUVs that can act as extensions of conventional surface vessels and submarines, enabling surveillance over wide areas for extended periods of time at costs far less than could be achieved with conventional surface vessels, submarines, and aircraft. These developments will rely on improvements in acoustic communications, energy systems, sensors, and onboard intelligence that will likely find their way into commercial and scientific practice.

43.4 Conclusions and Further Reading

The underwater environment is extremely hostile for human engineering activities. In addition to high pressures and hydrodynamic forces that are both nonlinear and unpredictable, water is not an appropriate media for electromagnetic communication except at short ranges. This pushes underwater technology to rely on acoustic communication and positioning systems that are characterized by low bandwidth. On the other hand, the ocean is extremely important for numerous human activities from the commercial, cultural, and environmental points of view.

Research on underwater robotic applications is active both from the technological and methodological aspects. The power endurance of commercially AUVs is currently up to 50 h; this will increase as energy-storage devices improve. Improved energy and power capability will enable longer missions, higher speeds, or better/additional sensors such as, e.g., more powerful lighting for underwater video/photography. The current trend for the price of AUVs prices is downward, with more and smaller research institutions building or buying AUVs to enrich their research results; moreover the setup of multiple-AUV systems is becoming cost effective. The goal is to develop fully autonomous, reliable, robust, decision-making AUVs.

There are a number of technology issues that are needed in order to improve AUV capabilities: to increase the underwater bandwidth of current acoustic modems, to increase onboard power to handle larger tools and interact more strongly with the environment, to create AUVs with significant hovering capability to allow better interaction, and to enable easier launch and recovery.

In the near future, the ROV/AUV dichotomy will likely become less prominent, with a variety of systems appearing that have attributes of both systems:

- For offshore oil and gas intervention tasks, a vehicle could transit to the work site as a self-powered, fully autonomous vehicle, then dock to the work site. Utilizing energy and communications infrastructure at the work site, the vehicle could then be operated much like a conventional ROV.

- Battery-operated ROVs can communicate with the surface by very lightweight fiber-optic links, enabling the mobility of an AUV but with a high-bandwidth connection to skilled human operators for complex intervention or scientific sampling tasks.
- Acoustic and optical data links can provide moderate to high communication bandwidths over short ranges, enabling human supervision without any tether restrictions. At longer ranges, more modest acoustic bandwidths are available.

These developments make marine robotics a challenging engineering problem with strong connections to several engineering domains. Sending an autonomous vehicle into an unknown and unstructured environment with limited online communication requires some onboard intelligence and the ability for the vehicle to react in a reliable way to unexpected situations [43.67, 68].

A major challenge concerning underwater robotics is the interaction with the environment by means of one or more manipulators. Autonomous UVMSs are still the object of research; the current trend is in developing the first semiautonomous robotic devices, which might be acoustically operated; moreover, if physically possible, the capability to dock to the structure where the intervention is needed might significantly simplify the control. The final aim might be to develop a completely autonomous UVMS, able to localize the intervention site, recognize the task to be performed, and act on it without docking to the station and without human intervention. This might make it possible to perform missions that are currently impossible such as autonomous archaeological intervention at deep sites. This would also enable the oil and gas industry to significantly decrease costs and risks to humans.

For further reading on the topic of underwater systems, the reader is referred to several survey articles, including [43.5, 26, 32, 33, 60]. Additionally, several journals cover oceanic engineering topics, including robotics aspects. A variety of symposia and workshops have been held on a regular basis. Some books/monographs treating marine robotics are [43.3, 7, 8, 16, 50].

References

- 43.1 J. Yuh, S.K. Choi, C. Ikehara, G.H. Kim, G. McMurty, M. Ghasemi-Nejhad, N. Sarkar N., K. Sugihara: De-

sign of a semi-autonomous underwater vehicle for intervention missions (SAUVIM), (1998) pp. 63–68

- 43.2 P. Marty: ALIVE: An autonomous light intervention vehicle, Advances in Technology for Underwater Vehicles Conference, Oceanology Int. (2004)
- 43.3 T.I. Fossen: *Marine Control Systems, Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles* (Marine Cybernetics AS, Trondheim 2002)
- 43.4 S. Bennett: A brief history of automatic control, IEEE Contr. Syst. Mag. **16**(3), 17–25 (1996)
- 43.5 J. Yuh, M. West: Underwater robotics, J. Adv. Robot. **15**(5), 609–639 (2001)
- 43.6 SNAME: Marine engineers: nomenclature for treating the motion of a submerged body through a fluid, Techn. Res. Bull. **1–5**, (1950)
- 43.7 T.I. Fossen: *Guidance and Control of Ocean Vehicles* (Wiley, New York 1994)
- 43.8 O.M. Faltinsen: *Sea Loads on Ships and Offshore Structures* (Cambridge Univ Press, Cambridge 1990)
- 43.9 J. Yuh: Modeling and control of underwater robotic vehicles, IEEE Trans. Syst. Man Cybern. **20**, 1475–1483 (1990)
- 43.10 T.I. Fossen, A. Ross: Guidance and control of unmanned marine vehicles, IEEE Contr. Eng. Series (Peregrinus, Stevenage 2006) pp. 23–42
- 43.11 W.E. Cummins: The impulse response function and ship motions, Techn. Rep. **1661**, *David Taylor Model Basin, Hydromechanics Laboratory* (DTIC, Washington 1962)
- 43.12 T.F. Ogilvie: Recent progress towards the understanding and prediction of ship motions, 5th Symposium Naval Hydrodynamics (1964) pp. 3–79
- 43.13 T. Perez, T.I. Fossen: Time-domain models of marine surface vessels for simulation and control design based on seakeeping computations, 7th IFAC Conf. Manoeuvring Contr. Marine Craft (IFAC, Lisbon, 2006)
- 43.14 T.I. Fossen: A nonlinear unified state-space model for ship maneuvering and control in a seaway, J. Bifurc. Chaos **15**(9), 2717–2746 (2005)
- 43.15 M. Nahon: Determination of undersea vehicle hydrodynamic derivatives using the USAF Datcom, Proc. Oceans Conf. (Victoria 1993) pp. 283–288
- 43.16 J.N. Newman: *Marine Hydrodynamics* (MIT Press, Cambridge 1977)
- 43.17 A.J. Healey, D. Lienard: Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles, IEEE J. Oceanic Eng. **18**, 327–339 (1993)
- 43.18 B. Stevens, F. Lewis: *Aircraft Control and Simulations* (Wiley, New York 1992)
- 43.19 I. Schjøberg, T.I. Fossen: Modelling and control of underwater vehicle-manipulator systems, 3rd IFAC Conf. Manoeuvring Contr. Marine Craft (IFAC, Southampton 1994) pp. 45–57
- 43.20 E.A. de Barros, A. Pascoal, E. de Sea: Progress towards a method for predicting AUV derivatives, 7th IFAC Conf. Manoeuvring Contr. Marine Craft (IFAC, Lisbon 2006)
- 43.21 N.P. Fofonoff, R.C. Millard: *Algorithms for Computation of Fundamental Properties of Seawater*, 44th edn. (UNESCO Technical papers in marine science, Paris 1983)
- 43.22 R. Eustice, H. Singh, J.J. Leonard, M. Walter, R. Ballard: Visually navigating the RMS Titanic with SLAM information filters, Proc. Robot. Sci. Syst. (Cambridge 2005) pp. 57–64
- 43.23 D.A. Smallwood, L.L. Whitcomb: Adaptive identification of dynamically positioned underwater robotic vehicles, IEEE Trans. Contr. Syst. Technol. **11**(4), 505–515 (2003)
- 43.24 S. Zhao, J. Yuh: Experimental study on advanced underwater robot control, IEEE Trans. Robot. **21**(4), 695–703 (2005)
- 43.25 S. Majumder, S. Scheding, H.F. Durrant-Whyte: Multisensor data fusion for underwater navigation, Robot. Autonom. Syst. **35**(2), 97–108 (2001)
- 43.26 J.C. Kinsey, R.M. Eustice, L.L. Whitcomb: A survey of underwater vehicle navigation: recent advances and new challenges, 7th IFAC Conf. Manoeuvring Contr. Marine Craft (IFAC, Lisbon 2006)
- 43.27 W.P.A. Van Lammeren, J. van Manen, M.W.C. Oosterveld: The Wageningen B-screw series, Trans. SNAME **77**, 269–317 (1969)
- 43.28 D.R. Yoerger, J.G. Cooke, J.J. Slotine: The Influence of thruster dynamics on underwater vehicle behavior and their incorporation into control system design, IEEE J. Oceanic Eng. **15**, 167–178 (1990)
- 43.29 A.J. Healey, S.M. Rock, S. Cody, D. Miles, J.P. Brown: Toward an improved understanding of thruster dynamics for underwater vehicles, IEEE J. Oceanic Eng. **20**(4), 354–361 (1995)
- 43.30 L. Bachmayer, L.L. Whitcomb, M.A. Grosenbaugh: An accurate four-quadrant nonlinear dynamical model for marine thrusters: theory and experimental validation, IEEE J. Oceanic Eng. **25**, 146–159 (2000)
- 43.31 T.I. Fossen, M. Blanke: Nonlinear output feedback control of underwater vehicle propellers using feedback from estimated axial flow velocity, IEEE J. Oceanic Eng. **25**(2), 241–255 (2000)
- 43.32 T.I. Fossen, T.I. Johansen: A survey of control allocation methods for ships and underwater vehicles, 14th IEEE Mediterranean Conf. Contr. Autom. (Ancona 2006) pp. 1–6
- 43.33 K.P. Valavanis, D. Gracanin, M. Matijasevic, R. Kolluru: Control architecture for autonomous underwater vehicles, IEEE Contr. Syst. **17**, 48–64 (1997)
- 43.34 P. Oliveira, A. Pascoal, V. Silva, C. Silvestre: Mission control of the MARIUS AUV: system design, implementation, and sea trials, Int. J. Syst. Sci. **29**(10), 1065–1080 (1998)
- 43.35 D. Brutzman, M. Burns, M. Campbell, D. Davis, T. Healey, M. Holden, B. Leonhardt, D. Marco, D. McClarin, B. McGhee: NPS Phoenix AUV software integration and in-water testing, Autonomous

- Underwater Vehicle Technol. AUV'96 (1996) pp. 99–108
- 43.36 A.P. Aguiar, A.M. Pascoal: Dynamic positioning and way-point tracking of underactuated AUVs in the presence of ocean currents, 41st IEEE Conf. Decision Contr. (Las Vegas 2002) pp. 2105–2110
- 43.37 A.A. Bennett, J.J. Leonard: A behavior-based approach to adaptive feature detection and following with autonomous underwater vehicles, IEEE J. Oceanic Eng. **25**(2), 213–226 (2000)
- 43.38 M. Breivik, T.I. Fossen: Principles of guidance-based path following in 2D and 3D, 44th IEEE Conf. Decision Contr. 8th Eur. Contr. Conf. (Sevilla 2005)
- 43.39 F.A. Papoulias: Bifurcation analysis of line of sight vehicle guidance using sliding modes, Int. J. Bifurc. Chaos **1**(4), 849–865 (1991)
- 43.40 R. Rysdyk: UAV path following for constant line-of-sight, Proc. 2nd AIAA Unmanned Unlimited Syst. Technol. Operations – Aerospace (San Diego 2003)
- 43.41 M.D. Feezor, F.Y. Sorrel, P.R. Blankinship, J.G. Bellingham: Autonomous underwater vehicle homing/docking via electromagnetic guidance, IEEE J. Oceanic Eng. **26**(4), 515–521 (2001)
- 43.42 D. Wettergreen, A. Zelinsky, C. Gaskett: Autonomous guidance and control for an underwater robotic vehicle, Int. Conf. Field Service Robot. (Pittsburgh 1999)
- 43.43 G. Antonelli, S. Chiaverini, R. Finotello, R. Schiavon: Real-time path planning and obstacle avoidance for RAIS: an autonomous underwater vehicle, IEEE J. Oceanic Eng. **26**(2), 216–227 (2001)
- 43.44 J.C. Hyland, F.J. Taylor: Mine avoidance techniques for underwater vehicles, IEEE J. Oceanic Eng. **18**, 340–350 (1993)
- 43.45 D.R. Yoerger, J.J. Slotine: Robust trajectory control of underwater vehicles, IEEE J. Oceanic Eng. **10**, 462–470 (1985)
- 43.46 R. Cristi, F.A. Pappulias, A. Healey: Adaptive sliding mode control of autonomous underwater vehicles in the dive plane, IEEE J. Oceanic Eng. **15**(3), 152–160 (1990)
- 43.47 D.B. Marco, A.J. Healey: Command, control and navigation experimental results with the NPS ARIES AUV, IEEE J. Oceanic Eng. **26**(4), 466–476 (2001)
- 43.48 M. Caccia, G. Veruggio: Guidance and control of a reconfigurable unmanned underwater vehicle, Cont. Eng. Prac. **8**(1), 21–37 (2000)
- 43.49 G. Antonelli, F. Caccavale, S. Chiaverini, G. Fusco: A novel adaptive control law for underwater vehicles, IEEE Trans. Contr. Syst. Technol. **11**(2), 221–232 (2003)
- 43.50 G. Antonelli: Underwater robots, Motion and force control of vehicle-manipulator systems. In: *Tracts in Advanced Robotics*, 2nd edn., ed. by G. Antonelli (Springer, Berlin, Heidelberg 2006)
- 43.51 S.B. Williams: A terrain aided tracking algorithm for marine systems, 4th Int. Conf. Field Service Robot. (2003) pp. 55–60
- 43.52 M. Dunbabin, P. Corke, G. Buskey: Low-cost vision-based AUV guidance system for reef navigation, IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp. 7–12
- 43.53 J.J. Leonard, H.J.S. Feder: Decoupled stochastic mapping, IEEE J. Oceanic Eng. **26**(4), 561–571 (2001)
- 43.54 S. Williams, G. Dissanayake, H. Durrant-Whyte: Towards terrain-aided navigation for underwater robotics, Adv. Robot. **15**(5), 533–549 (2001)
- 43.55 P. Newman, J. Leonard: Pure range-only sub-sea SLAM, IEEE Int. Conf. Robot. Autom. (Taipei 2003) pp. 1921–1926
- 43.56 T.W. McLain, S.M. Rock, M.J. Lee: Experiments in the coordinated control of an underwater arm/vehicle system, Auton. Robot. **3**(2), 213–232 (1996)
- 43.57 J.S. Ferguson, A. Pope, B. Butler, R. Verrall: Theseus AUV – two record breaking missions, Sea Technol. Mag. **40**, 65–70 (1999)
- 43.58 A. Alessandri, M. Caccia, G. Veruggio: Fault detection of actuator faults in unmanned underwater vehicles, Cont. Eng. Prac. **7**, 357–368 (1999)
- 43.59 K.C. Yang, J. Yuh, S.K. Choi: Fault-tolerant system design of an autonomous underwater vehicle – ODIN: an experimental study, Int. J. Syst. Sci. **30**(9), 1011–1019 (1999)
- 43.60 G. Antonelli: A survey of fault detection/tolerance strategies for AUVs and ROVs. Recent advances. In: *Fault Diagnosis and Tolerance for Mechatronic Systems*, Springer Tracts in Advanced Robotics, ed. by F. Caccavale, L. Villani (Springer, Berlin, Heidelberg 2002) pp. 109–127
- 43.61 T.K. Podder, G. Antonelli, N. Sarkar: An experimental investigation into the fault-tolerant control of an autonomous underwater vehicle, J. Adv. Robot. **15**, 501–520 (2001)
- 43.62 M. Caccia, R. Bono, G. Bruzzone, G. Bruzzone, E. Spirandelli, G. Veruggio: Experiences on actuator fault detection, diagnosis and accommodation for ROVs, Int. Symp. Unmanned Untethered Submersible Technol. (Durham, New Hampshire 2001)
- 43.63 E. Fiorelli, P. Bhatta, N.E. Leonard, I. Shulman: Adaptive sampling using feedback control of an autonomous underwater glider fleet, Int. Symp. Unmanned Untethered Submersible Technol. (Durham, New Hampshire 2003)
- 43.64 C.J. Cannell, D.J. Stilwell: A comparison of two approaches for adaptive sampling of environmental processes using autonomous underwater vehicles, Proc. Oceans Conf. (Brest 2005) pp. 1514–1521
- 43.65 S. Kalantar, U. Zimmer: Distributed shape control of homogeneous swarms of autonomous underwater vehicles, Auton. Robot. **22**(1), 37–53 (2006)

43.66

A. Pascoal, C. Silvestre, P. Oliveira: Vehicle and mission control of single and multiple autonomous marine robots. In: *Advances in Unmanned Marine Vehicles*, IEEE Control Engineering, ed. by G. Roberts, R. Sutton (Peregrinus, New York 2006) pp.353–386

43.67

J. Yuh: Exploring the mysterious underwater world with robots, 6th IFAC Conf. Manoeuvring Contr. Marine Craft (IFAC, Girona 2003)

43.68

T. Ura: Steps to intelligent AUVs, 6th IFAC Conf. Manoeuvring Contr. Marine Craft (IFAC, Girona 2003)

Aerial Robotics

Eric Feron, Eric N. Johnson

A wide array of potential applications exist for robots that have the level of mobility offered by flight. The military applications of aerial robotics have been recognized ever since the beginnings of powered flight, and they have already been realized to sometimes spectacular effect in surveillance, targeting, and even strike missions. The range of civilian applications is even greater and includes remote sensing, disaster response, image acquisition, surveillance, transportation, and delivery of goods.

This chapter first presents a brief history of aerial robotics. It then continues by describing the range of possible and actual applications of aerial robotics. The list of current challenges to aerial robotics is then described. Building from basic notions of flight, propulsion, and available sensor technology, the chapter then moves on to describe some of the current research efforts aimed at addressing the various challenges faced by aerial robots.

The challenges faced by aerial robots span several and distinct fields, including state regulations, man-machine interface design issues, navigation, safety/reliability, collision prevention, and take-off/landing techniques. The size of aerial robots can considerably influence their flight dynamics, and small aerial robots can end up looking considerably different from their larger counterparts. Similar to their manned counterparts, aerial robots may enjoy diverse propulsion systems and operate over large speed ranges.

Aerial robots must be equipped with reliable position and actuation equipment so as to be capable of controlled flight, and this constitutes a nontrivial requirement prior to doing research

or development in this field. However, many universities, research centers, and industries have now met this requirement and are actively working on the challenges presented above. The largest obstacle to the commercial development of aerial robots is, however, the necessity to comply with and support a regulatory environment which is only beginning to address these rapidly developing systems.

44.1 Background	1010
44.2 History of Aerial Robotics	1010
44.3 Applications of Aerial Robotics	1012
44.3.1 Possible Applications of Aerial Robots	1012
44.3.2 Current Applications	1013
44.4 Current Challenges	1014
44.4.1 Regulations and Certification	1014
44.4.2 Human-Machine Interfaces	1014
44.4.3 Navigation	1015
44.4.4 Agile Flight and Fault Tolerance	1015
44.4.5 Obstacle Avoidance	1015
44.4.6 Aerial Robot Landing and Interaction with Other Vehicles	1015
44.4.7 Multivehicle Coordination	1015
44.5 Basic Aerial Robot Flight Concepts	1015
44.5.1 Aerial Robot Flight and the Importance of Scales	1015
44.5.2 Propulsion Systems	1017
44.5.3 Flight Vehicle Types and Flight Regimes	1018
44.5.4 Lighter-Than-Air Systems	1019
44.6 The Entry Level for Aerial Robotics: Inner-Loop Control	1020
44.6.1 Sensing and Estimation	1020

44.6.2	Estimator Design	1021	44.7.4	Reactive Flight in Cluttered Environments and Obstacle Avoidance	1025
44.6.3	Inner-Loop Control.....	1021	44.7.5	Path Planning and Higher-Level Planning Capabilities.....	1025
44.7	Active Research Areas.....	1022	44.7.6	Integrated Aerial Robotic Operations: Aerial Robotics Contests	1025
44.7.1	Interfacing with the Human Infrastructure: Meeting the Regulations	1022	44.8	Conclusions and Further Reading.....	1026
44.7.2	High-Agility Flight.....	1023	References		1027
44.7.3	Take-Off, Landing, and Interaction with Other Vehicles	1023			

44.1 Background

The term *aerial robotics* is often attributed to *Robert Michelson* [44.1], as a way to capture a new class of highly intelligent, small flying machines. However, it is clear that the range of systems and activities covered under the label *aerial robotics* could extend much further, and that its roots can be found far back in the beginning of the 20th century, together with the birth of aviation. Behind the word *aerial robotics* we can find several meanings: it could mean *robotic flying machines*, that is, a mission-independent, platform-oriented concept; however, it could also mean *robotics that use flying machines*, that is, a platform-independent, mission-oriented con-

cept. Finally, it could mean a combination of the above, that is a description of the robotic platform, together with its robotic mission. In aerospace jargon, robotic flying machines are commonly referred to as *unmanned aerial vehicles (UAVs)*, while the entire infrastructures, systems and, human components required to operate such machines for a given operational goal are often called *unmanned aeral systems (UASs)*. Finally, it is worth noting that many current manned aerial systems definitely carry relevant features of some robotic systems, and so much of this discussion is relevant to manned aircraft.

44.2 History of Aerial Robotics

The history of aerial robotics is very closely tied to the history of flight itself. Indeed, the rate of fatalities associated with early manned flight tests probably convinced engineers that there was a need to operate flying machines without the presence of humans on board even before potential applications of unmanned aircraft surfaced. In 1903, heavier-than-air flight was unambiguously shown to be feasible, following the achievements of the Wright brothers. The first successful powered flight was unmanned, presumably to reduce the risk to the pilot and to allow a smaller and less expensive vehicle (reasoning that is still put forth today) by Samuel P. Langley’s *Number 5* in 1896 [44.2].

Finding a truly defining moment for aerial robotics is a challenge, with encyclopedias dating the concept back to Leonardo da Vinci, while *Newcome* [44.3], in his history of unmanned aviation, gives early credit to Nikola Tesla for devising a robotic vehicle remotely controlled

by electromagnetic waves, and with enough onboard logic to recognize and execute remotely transmitted orders. However, the concept imagined and engineered by Tesla did not apply specifically to airborne vehicles.

Using the definition

An aerial robot is a system capable of sustained flight with no direct human control and able to perform a specific task,

leads us almost immediately to the Hewitt–Sperry automatic airplane, developed before and during World War I [44.3]. The airplane’s purpose was to act as a flying torpedo, carrying onboard *intelligence* to sustain flight over long periods of time without human intervention. Such intelligence was provided by a complex system involving Sperry’s own gyroscopes, mechanically coupled to the airplane’s control surfaces so as to stabilize the vehicle. This made the airplane suitable for

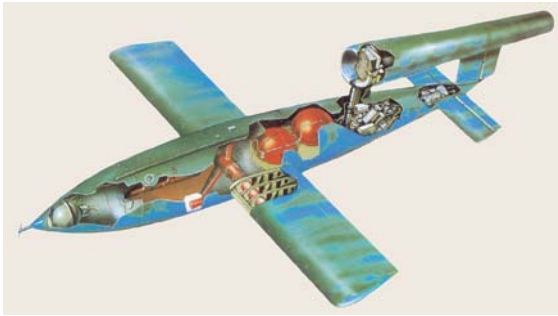


Fig. 44.1 V-1 German cruise missile (1940s)



Fig. 44.2 QH-50 DASH unmanned helicopter on final approach (US Navy)

human remote control and, eventually, prosecution of distant targets. As discussed later, one of the key characteristics of aerial robotics is this particular necessity for the robot to sustain itself in the air with no human intervention, which requires the early adoption and understanding of the critical role played by *onboard intelligence*, much more so than other robotic applications. While a string of inventors in many countries came to develop ever more sophisticated machines, credit goes to the German V-1 cruise missile for making a lasting, and unfortunately deadly, impact on large segments of the population in England. This form of robotic aircraft owed its relative inefficiency (three out of four vehicles reportedly missed their target – predominantly London) to mechanical failures and lack of good navigation capabilities beyond dead-reckoning assisted by gyroscopic

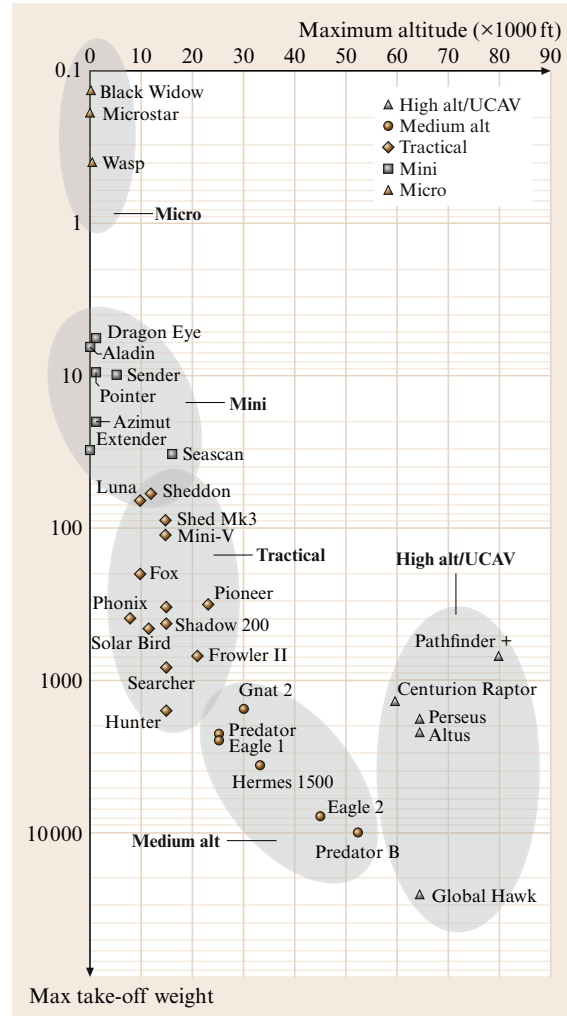


Fig. 44.3 Taxonomy of unmanned aerial vehicles (after R. Weibel [44.4,5])

devices. Many of the ensuing aerial robotics developments followed the initial idea of defense applications for unmanned systems, that is, ever more accurate flying machines for the purpose of either reconnaissance or weapon delivery. One notable machine was the US Navy's Gyrodyne QH-50 DASH, an unmanned helicopter developed in the 1950s and operated from US destroyers, which was able to perform reconnaissance missions and deliver torpedoes (Fig. 44.2). However, these machines remained relatively *unintelligent*, and their level of autonomy remained limited to the ability to sustain flight using complex inertial and other measurement systems.

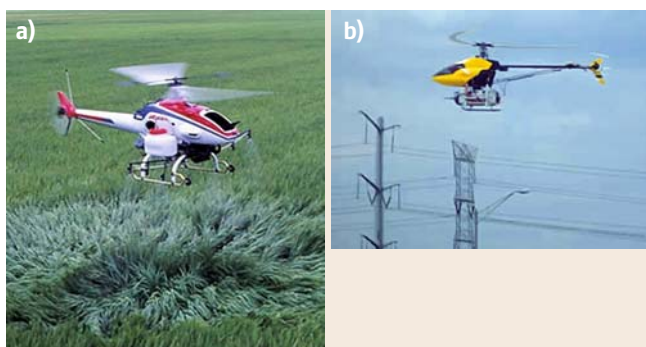


Fig. 44.4a,b Autonomous helicopters: (a) Yamaha's R-MAX (b) Nascent Technologies Corp's XS

From a robotics perspective, probably the next significant technological enabler is the advent of lightweight processors and sensor systems, together with global navigation satellite systems, which allowed aerial robots to perform increasingly complex tasks. Japan, motivated by a policy of food self-sufficiency combined with a massive shortage of agricultural workforce, took a lead in aerial robotics by developing highly reliable helicopters in the 1980s, such as the Yamaha R-50 and subsequent Yamaha R-Max (Fig. 44.4), with similar systems developed by other companies such as Yanmar. These robotic helicopters are used primarily for crop dusting applications, especially over wet rice fields. These vehicles also turned out to be very popular among universities and other institutions for their unmatched

ability to fly reliably and take off and land from limited areas, allowing researchers to focus their attention on developing higher levels of autonomy beyond basic vehicle navigation and control. Military applications of unmanned robotics followed the track of the German V1, with the advent of modern cruise missile technology.

From the mid 1980s on, the development of aerial robots has followed an exponential pace, with one notable trend for operational aerial robots to systematically find military applications as their most significant market.

A snapshot of available aerial vehicle platforms and systems appeared recently in [44.6], and regular updates on available platforms can be found in the aerospace literature. The chart in Fig. 44.3 illustrates the number of machines currently under development or in operation, which exceeds 200 vehicle types. From this chart, one concludes, however, that the vast majority of current, operational aerial robots are fixed-wing aircraft, and that they tend to be present at all altitudes.

We must also remark that onboard robotic intelligence has made its way not only into unmanned aircraft but also manned aircraft. Many commercial airliners now have the ability to fly automatically from right after take-off (a decision left to the pilot) to right after landing, by engaging the autopilot and letting the aircraft fly a predetermined profile. In addition, these vehicles are now able to make systems-management decisions based on sensor inputs, sometimes escaping the human pilot's ability to understand them.

44.3 Applications of Aerial Robotics

Listing all possible applications of aerial robotics is very challenging. However, there are fewer actual implementations, because of the necessity for the corresponding operations to comply with stringent air safety regulations. In the following, a brief description of possible and current applications is provided.

44.3.1 Possible Applications of Aerial Robots

The list of possible applications of aerial robots is long. According to [44.6,7], such applications fall within nine categories:

- *Remote sensing* such as pipeline spotting, powerline monitoring, volcanic sampling, mapping, meteorol-

ogy, geology, and agriculture [44.8,9], as well as unexploded mine detection [44.10].

- *Disaster response* such as chemical sensing, flood monitoring, and wildfire management.
- *Surveillance* such as law enforcement, traffic monitoring, coastal and maritime patrol, and border patrols [44.11].
- *Search and rescue* in low-density or hard-to-reach areas.
- *Transportation* including small and large cargo transport, and possibly passenger transport.
- *Communications* as permanent or ad hoc communication relays for voice and data transmission, as well as broadcast units for television or radio.
- *Payload delivery* e.g., firefighting or crop dusting.
- *Image acquisition* for cinematography and real-time entertainment.

Military applications of aerial robots follow the same descriptive lines, with a particular emphasis on remote sensing of humans and critical infrastructure, surveillance of human activity, and payload delivery (bombs, missiles, and ad hoc ground infrastructures devoted to communication and surveillance).

44.3.2 Current Applications

Current applications of aerial robots are somewhat fewer and they are at present driven by the military context.

Aerial Observations

The most important application of aerial robots is aerial observations, which can then be used for terrain mapping, environmental surveys, crop monitoring, target identification etc. There is a divide, however, between the state of the art for military applications and civilian applications, detailed below.

Military Operations. Military and government use of aerial robots has sharply increased in recent years in war zones. As a result, dozens of vehicles are now delivered every month, and end up flying in “hot” areas around the globe, most notably in Southwest Asia. The machines being flown range from man-portable machines flying at low altitudes, such as the Pointer or Raven aircraft, to mid-sized machines such as the Aerosonde, Seascan, or Shadow unmanned vehicles, to larger-sized vehicles such as the Predator or Global Hawk. Their wings span from a meter or so for the smaller vehicles to 35 m for Global Hawk (the same as a Boeing 737). Besides their use in military areas or war zones, these machines now find applications in border surveillance, with a particular interest in oceanic borders, where vehicles operate in desert or quasi-desert areas.

Civilian and Private Applications. Current civilian applications of aerial robots for surveillance and observation remain sporadic and ad hoc: unlike many other robotic devices, civilian aerial robots do not operate in *closed* environments but in civilian airspace, which is subject to strong safety regulations that do not yet systematically accommodate aerial robots. Consequently, the current trends in civilian aerial robotics are as follows.

Small-scale, intermittent civilian aerial robotic applications tend to happen in relatively isolated environments (e.g., for film making or environmental surveys), and often follow the safety and operations rules most familiar to their operators, derived

from model aircraft operations. Most often, the operated machines do in fact bear much resemblance to radio-controlled model airplanes. Other intermittent applications involve the use of unmanned vehicles for specific reconnaissance tasks, such as the detection of fish banks from trawlers [44.12]. Such a task constituted one of the original purposes for the development of machines such as the Seascan unmanned aerial vehicle.

Long-term scientific applications such as atmospheric sampling experiments [44.13] appear to benefit considerably from aerial robots. One report [44.13] reads

From March 6 to March 31 2006, we probed the polluted atmosphere over the North Indian Ocean with lightweight unmanned aerial vehicles (or UAVs) fully equipped with instruments. This UAV campaign launched from the Maldives laid a solid foundation for the use of UAVs to study how human beings are polluting the atmosphere and their impact on climate, including global warming.

Because such activities naturally require much planning ahead, special permits can be obtained from aviation authorities within time limits that do not significantly affect the overall experimental project. Other scientific missions led with success include [44.14], where the authors were able to survey Mount St. Helens (then active) by taking advantage of the temporary interdiction to fly in the vicinity of the volcano.

With the progressive introduction of aerial robots in the regulatory framework of many countries, we believe that intermittent applications of aerial robotics in populated areas will eventually become commonplace. However, this requires that flight authorizations be delivered within a fraction of the time duration of the event: for example, firefighting operations are often triggered within a few seconds of the fire alert. Permits for aerial robotic support should therefore be delivered about as quickly if they are ever to be embraced by firefighters.

Payload Delivery

Under the heading *payload delivery*, we find the numerous applications of aerial robots aimed at delivering solid, liquid or gaseous products in areas that are hard to reach for humans. So far, the most successful civilian application has been chemical crop spraying using small unmanned helicopters. Leveraging the high costs and prices associated with crop culture in Japan, several thousand helicopters have been purchased by farmers, resulting in a profitable

operation both for themselves and for the helicopter manufacturers, among them Yamaha and Yanmar. However, this application remains unique and involved the involvement of Japan's government for it to be successful.

Besides this particular application, military applications form the bulk of unmanned aerial robotics for the purpose of payload delivery, beginning in its crudest form with missiles, and evolving towards cruise missiles, able to navigate for thousands of miles and reach their targets with high precision. One of the most talked-about recent military application of aerial robots for payload delivery involves the Predator aircraft equipped with Hellfire missiles.

44.4 Current Challenges

In the following, we introduce six major challenges for aerial robotics. This list is not meant to be exhaustive, but it reflects the current focus of researchers. How these challenges are addressed will be discussed later.

44.4.1 Regulations and Certification

A big challenge to the success of aerial robots is doubtlessly their acceptance by certification authorities. Indeed, the operation of aerial robots is currently significantly limited by regulatory constraints. This is due to the complex set of regulations put in place by national agencies (e.g., the Federal Aviation Administration in the US, the National Air Traffic Services in the UK, or the Direction Générale de l'Aviation Civile in France), whose aim is to maintain very high levels of safety for air traffic. The downside of the excellent safety record reached by regulatory agencies is their (justified) risk adversity, and therefore a slow acceptance of disruptive technologies such as aerial robots. This is compounded by the current rapid pace of change and related lack of standards among aerial robotic systems and how they are used. However, with the help of other organizations, such as the Radio Technical Commission for Aeronautics (RTCA), regulatory agencies have moved forward towards establishing rules for the routine operation of aerial robots. Such rules include the ability for aerial robots to *see and avoid* or *sense and avoid* other traffic at least as well as a human pilot.

In the recent past, many aerial robotics research activities and corresponding flight tests have occurred at very low altitude. In the absence of clearly defined

General Characteristics of Current Applications: Level of Autonomy

Most, although not all, aerial robots currently under operation are automatically controlled as far as their dynamics are concerned. However, higher levels of autonomy, such as path planning, object detection, and recognition and mission management involve human operators, who always remain in contact with the flying machine. Thus not much distinguishes current aerial robots from traditional manned aircraft, except that the pilot sits on the ground rather than in the air.

As such, most of today's operational aerial robots may be justifiably called remotely piloted vehicles (RPVs).

rules by regulatory bodies until recently (2000), many researchers have operated under the rules of local radio-controlled aircraft associations (e.g., the Academy of Model Aeronautics). However, there is a rapidly growing trend for radio-controlled vehicles to incorporate more onboard electronics, including radio transmitters and sometimes guidance systems. In this environment, one can expect regulatory bodies such as the FAA to continue to evolve their policies.

The ensuing challenge is for the research community is to develop the requirements and subsequent technology that meets the constraints set by the regulatory agencies, or to propose and justify alternate constraints. In particular, the maturation of aerial robots leading to their everyday use in populated areas will require the development of more reliable components, defined maintenance procedures, formal training programs, and the automation of emergency procedures (such as the forced landing process). The core technology for UAS already exists to demonstrate safety concepts. However, developing highly dependable systems – and making such dependability guarantees acceptable to the regulatory authorities – is a current and urgent challenge.

44.4.2 Human–Machine Interfaces

The pilot interfaces used for manned aircraft have evolved continuously since the first manned aircraft. The standards that exist today directly benefit safety and operator costs by minimizing operational errors and training time when transitioning between aircraft types. Despite ongoing development efforts [44.15], this can-

not be said of aerial robotic systems, which run cover a much wider range of autonomy, mission capability, and operator skill. Add to this the desire to have single operators control multiple aircraft, and it clear this area presents an ongoing challenge for researchers.

44.4.3 Navigation

Figuring out absolute and relative position is a central issue for aerial robots, as it is for other robotics activities. The existence of a significant manmade infrastructure (the global navigation satellite system – GNSS) makes basic navigation easy but remains the subject of an intense debate; indeed, systems that overly depend on such infrastructure lack resilience and tolerance to positioning services shortage, whether such a shortage originates from the system itself or from the particular robot configuration (in cluttered environments such as cities). This situation will improve with the development of highly reliable multimode navigation systems with built-in integrity monitors, and with three independent satellite navigation constellations (Glonass, Galileo, and GPS) currently deployed or under deployment.

The challenge for researchers is to develop navigation technologies that allow aerial robots to live without manmade external navigation infrastructure, to handle the situations when it is not available.

44.4.4 Agile Flight and Fault Tolerance

Nearly every aircraft in operational use today has been challenged to fly far beyond its flight envelope during flight tests, including famous maneuvers such as that of the Boeing 707 that, during early flight demonstrations to customers, performed a full barrel roll. The purpose of these demonstrations is not only to show the full capabilities of the vehicle, but also to bring a sense of safety to the pilots, that the aircraft is still able to perform well after its goes into some upset condition. What applies to large, manned aircraft also applies to aerial robots,

which must be able to keep operating well at unusual attitudes and under partial system failures such as loss of actuation [44.16]. Researchers must develop automation systems that meet this need.

44.4.5 Obstacle Avoidance

The ability for a vehicle to manage its position away from obstacles represents a significant issue and a necessity for low-altitude operations in crowded environments. One of the key features of aerial robots is their possibly high speeds, which challenges many existing sensor management and data processing algorithms, especially their ability to detect hard-to-see obstacles such as suspended cables quickly.

44.4.6 Aerial Robot Landing and Interaction with Other Vehicles

Owing to the finite endurance of aerial robots, landing and docking are particularly important to them. While landing constitutes an important element, docking with other vehicles, such as during aerial refueling, is also very important. All operations involving close coordination and physical interaction between vehicles or between a vehicle and the ground require further research.

44.4.7 Multivehicle Coordination

Several tasks require aerial robots to operate as a group, rather than as individual systems. This happens, for example, in order to create phased array antennas, or to perform object geolocation, or to improve the quality of a surveillance service (e.g., fire monitoring). Other tasks requiring multivehicle coordination include the requirement for collision avoidance. More recently, multivehicle coordination has been seen as a valuable way to design aerial robotic systems that remain functional despite individual vehicle failures.

44.5 Basic Aerial Robot Flight Concepts

44.5.1 Aerial Robot Flight and the Importance of Scales

Like all flying machines, the performance of aerial robots depends extensively on: (1) their size and (2) the characteristics of their lifting mechanisms (wings, ro-

tors). A detailed description of vehicle flight mechanics is outside of the scope of this chapter; we can, nevertheless, recall a few fundamental and useful notions critical to successful flight. The reference [44.17] is an excellent and entertaining introduction to the subject, while [44.18, 19] offer a more academic perspective on



Fig. 44.5 Flying wing (Northrop's YB47) and its shrunk version flying together

the matter. One important quantity is the *mass* of a flying machine. Roughly speaking, the mass of a flying machine is proportional to its volume, and therefore grows like the cubic power of its size. Another quantity is the lifting forces that keep a vehicle up in the air; these are proportional to the pressure exercised on the lifting surface (rotor or wing), times the *area* of the lifting surface, that is, roughly the second power of the vehicle size. The pressure itself is proportional to the density of the surrounding atmosphere (it need not be air only, think of Mars), multiplied by the square of the average velocity of the gas molecules relative to the lifting surface.

For illustrative purposes, consider the flying wing shown in Fig. 44.5 and a notional scaled-down version of it flying together. To make matters simpler, we assume that the scaled-down wing is about half the size of the full-sized wing. We now examine the impact of scales on the way these wings must fly.

Consider for example the lift created by the full-scale flying wing depicted in Fig. 44.5: it is proportional to $S\rho V^2\alpha$, where S is its total surface, ρ is the air density, V is the wing speed relative to the surrounding air, and α is the angle of attack (roughly speaking the angle between the wing chord and the flow of air).

To get an idea of the importance of scales, and following arguments developed in much greater detail in [44.17], we now examine the requirements for the scaled-down wing to fly at the same speed as the large wing, assuming all its components are shrunk by a factor two in size as shown in the picture, and examine the consequences of having to meet such requirements.

First, the mass of the wing roughly gets divided by a factor 8 (2^3). However, its lifting surface has shrunk by

a factor of 4 only (2^2). So, if we were to fly this smaller wing at the same speed, same altitude, and same angle of attack as its big sister, the total generated lift must be $S/2\rho V^2$, that is, twice as much as necessary to balance out the effect of gravity.

Several solutions to this issue are possible: to reduce the the actual wing dimensions at constant mass, slow it down, or reduce its angle of attack.

Shrink the Wing

To obtain the proper lift (while keeping the speed and angle of attack constant), we must shrink the wing area by another factor of two, or the wing dimensions by a factor $\sqrt{2}$. Thus we already see one important conclusion, which is that, at equal speed and angle of attack, *the relative size of the wings with respect to the overall vehicle must shrink as the overall vehicle size goes down*. Borrowing again from [44.17], this explains much of why a Boeing 747 looks, with its large deployed wings and relatively narrow fuselage, like a condor while a B737 feels more like a puffin, and the smaller Embraer 145 is like a dart, as shown in Fig. 44.6. All

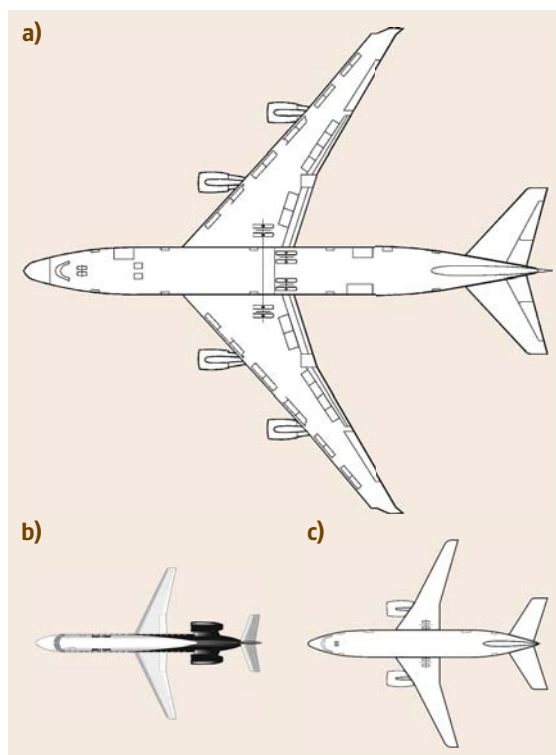


Fig. 44.6a-c Relative fuselage and wing sizes for various aircraft: (a) Boeing 747: (b) Embraer 145: (c) Boeing 737

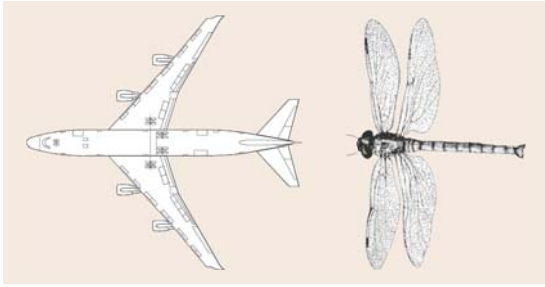


Fig. 44.7 Boeing 747 and dragonfly

these aircraft can fly about the same speeds and altitude ranges.

Reduce the Speed

If this option is chosen, then speed must be divided by a factor $\sqrt{2}$ for our scaled model to balance lift and weight. The consequences of reducing speed are many: the time required for mission completion of course increases. On the other hand, the drag generated by the flying machine (and which must be paid for by the propulsion system) goes down.

Pushed to their limits, the consequences of slowing down the vehicle as it shrinks can be quite dramatic: consider a dragonfly (one of the role models for micro-aerial robots) trying to land next to a Boeing 747 at the same airport. The figure below shows that the two share (very roughly) the same proportions.

For the sake of simplicity, assume that the dragonfly is the 1/1000 scaled-down version of the Boeing 747. In order for both to fly level, and according to our rule, the dragonfly must fly a factor $\sqrt{1000} = 32$ slower than the B747. Assume the B747 flies at 500 km/h; that makes the dragonfly fly at about 17 km/h. Imagine now that the weather is gusty, with winds topping 30 km/h. The 747 (and its passengers) will see little variation in airspeed (from 470 to 530 km/h), and the variation in produced lift will be 33%, enough to shake the aircraft a bit, but not unusually bad. As for the dragonfly, the same gusts will create airspeed variations of well over 100%, and the produced lift will vary from zero to five or six times the nominal lift. A rough ride naturally follows, and indeed, the flight of smaller vehicles often looks much less smooth than that of large ones.

Reduce the Angle of Attack

The latter option, reducing the angle of attack, rests upon the fact that, roughly, the lift created by a wing (or a rotor), is a linear function of the angle of at-

tack. This makes it possible to fly about the same speed with a scaled-down model of a flying machine. However, this option comes with significant drawbacks, especially for fixed-wing aircraft. In particular, the *sensitivity* of the lift created by the wing to external perturbations (e.g., air turbulence and wind gusts) would again be higher, creating another recipe for *bumpy rides*.

The previous considerations about the forces acting on aerial vehicles also apply to moments: consider the flying wings shown in Fig. 44.5, and assume that their density (mass per unit volume) is constant throughout. Their angular inertia about any axis are proportional to the *fifth* power of their size. On the other hand, the forces that apply to the wings are proportional to their area; thus when *moments* are computed, forces are multiplied by distances, and the resulting moments become proportional to *volume*, that is, the third power of vehicle size. Consider then the angular momentum equation

$$J\ddot{\theta} = M, \quad (44.1)$$

where J is the moment of inertia of the vehicle and M is the applied torque. The term to the left of (44.1) decreases *much faster* with vehicle size than that to the right of the equation. As a consequence, we might immediately conclude that the scaled-down flying wing is inherently *much more maneuverable* than the larger one, in the sense that it can change orientation much faster.

This opens up a wealth of possibilities for robotics: venturing into the world of small flying robots, enabled by improvements of battery power and computation densities opens new possibilities in terms of defining the way these vehicles fly and interact with their environment.

44.5.2 Propulsion Systems

Several propulsion systems exist for aerial robots, including: jet, internal combustion, rocket, and electric. Older but recurrent options also include pulse engines such as those used on the German V1.

Owing to established aircraft and helicopter propulsion technologies, internal combustion engines and jet engines form the bulk of the propulsion means for medium to large-sized operational vehicles (50 kg or more), allowing many of them to fly reliably over periods of several hours to several tens of hours. When considering operational robots, the kind of fuel used matters: preference is given to fuels al-

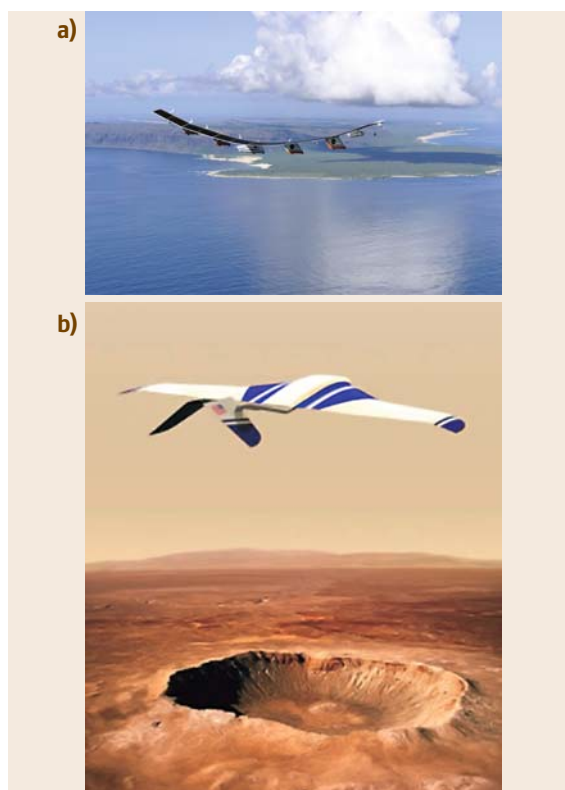


Fig. 44.8 (a) Helios high-altitude long-endurance aircraft, (b) Mars aircraft [44.20] (Source: NASA)

ready used in other devices, and preference goes to heavy fuels, which are less prone to sudden and dangerous combustion or explosion, for example after a crash.

Electric propulsion systems, once unthinkable, have become a reality for several small-sized aerial robots, thanks to the development of affordable brushless electric engines and lightweight batteries. Initially developed for computer and communication applications, these batteries have been very quickly adopted by small-sized (a few kg) aerial robots such as Aerovironment's Pointer and Raven aircraft, which are able to fly over periods exceeding one hour. The National Aeronautics and Space Administration's (NASA) Pathfinder unmanned aircraft combines lightweight electric engines with wing-mounted solar panels to yield the aircraft shown in Fig. 44.8.

A notable departure from these propulsion systems is the Mars airplane's propulsion system [44.20]: with an inert, low-density atmosphere on Mars, such a vehicle relies on a rocket engine for propulsion.

44.5.3 Flight Vehicle Types and Flight Regimes

Several vehicle types form the bulk of aerial robots, including fixed-wing machines, helicopters, flapping wing systems, and combinations thereof. The boundaries between these vehicle types are, however, mostly inherited from historical developments and intellectual stove-piping, rather than any fundamental guidelines dictated by the laws of mechanics and thermodynamics. For that reason, it is easier and more logical to introduce different *flight regimes* than flight vehicles, although to every regime there naturally corresponds one particular vehicle.

There are essentially two flight regimes. In the first regime, called *hover*, the speed of the vehicle relative to the surrounding air is small, such that few or no forces act on the vehicle except those resulting from the propulsion system itself. In the second regime, which we may call *cruising flight*, there is a significant relative speed between the vehicle and its surrounding environment, and significant aerodynamic forces act on the vehicle; these aerodynamic forces then largely dominate those generated by the power system.

Hover

Hover is the condition when the vehicle body does not move significantly with respect to the air mass surrounding it. Under these conditions, only propulsion systems are available to keep the vehicle up in the air (for heavier-than-air systems). Helicopters epitomize these situations, and they are especially designed to sustain such hover conditions over long periods of time. Helicopters come in all sizes and shapes. Robotic helicopters are best represented by Yamaha's R-50, and now RMAX models (see, for example, Fig. 44.4), and both have been a staple of airborne robotics research for years at several academic institutions because of their reliability and available payload, which allows them to carry many instruments of interest to robotics research (including navigation sensors such as video cameras, laser range finders, and radars). With the evolution of the economic and political context, one is bound to see other such machines abound in the future. Indeed, the ability to hover is extremely useful for delivery/pickup of materials, rescue missions, and, in general, any operations that require close proximity to rugged terrain.

Helicopters are not the only vehicles capable of hover, see for example the hover-capable fixed-wing aircraft in Fig. 44.9. Hovering aircraft, such as tailsitters,

have been tested successfully since the 1950s at the very least, and it is a classic trick for experienced remote control pilots to hover airplanes. Transitions from hover to forward flight and back have been automated [44.21]. As radio-control (R/C) equipment shrinks in size and mass, new generations of hovering vehicles will become available. Some of these vehicles include micro air and flapping wing vehicles.

Hovering flight typically not very fuel inefficient: the fuel consumption of a hovering vehicle can exceed that of a fixed-wing vehicle by an order of magnitude or more. This kind of consideration has led manufacturers to seek some of the mixed configurations shown in Fig. 44.9.

Cruising Flight

During cruising flight the aerial robot mostly uses its available surfaces and its speed relative to the surrounding atmosphere to generate lift and maintain altitude. Unlike hovering flight, cruising flight usually results in the aerial robot constantly meeting *fresh air*, which makes the range of adverse events to flight quite narrower. This, of course, is not true in the case when aerial robots fly in formation, in which case turbulence created by one robot may affect its neighbor(s), sometime adversely, and sometimes positively [44.22].

Robotic airplanes such as those shown in Fig. 44.8 epitomize fuel-efficient cruising flight, with large and highly optimized wings. Both aircraft are part of current NASA programs. While optimized for flight, the wings of the Mars aircraft must also be optimized for tight packaging and deployment constraints at the end of its long trip from Earth to Mars.

While many fixed-wing systems are optimized for cruising flight, any system in forward flight operates according to the same principles; for example, a helicopter in forward flight operates like an airplane whose wing is a flat disc spanning the area covered by its rotor. Constant-velocity cruising flight that generates lift from the available aerodynamic surfaces is more fuel efficient than hovering.

Stalled and High-Angle-of-Attack Flight

This flight condition can be seen as a *transitional* flight condition, where the characteristics of both forward flight and hover are present. Typically, an aircraft stalls when it tries to maintain altitude at low speeds: flying level at lower speeds forces the aircraft's angle-of-attack to increase for the wings to produce more lift. However, past a critical angle of attack, the trend reverses



Fig. 44.9a–c Non-helicopter, hover-capable vehicles: (a) Joint Strike Fighter, (b) 1950s tailsitter aircraft and (c) Aurora Flight Sciences' Golden Eye 100

and the lift produced by the wing *decreases* as the angle of attack keeps increasing. This reduced aerodynamic lift must then be compensated by increased throttle, resulting in a situation where the aircraft propeller not only acts as a means to move the aircraft forward, but also directly participates in maintaining aircraft altitude.

This flight condition, whether experienced on a helicopter or airplane, often results in important changes of the effect of control mechanisms, for example, a stalled Piper Tomahawk trainer aircraft at low throttle setting will experience ineffective ailerons, while its rudder efficiency will shift from yaw axis to roll axis control [44.23].

44.5.4 Lighter-Than-Air Systems

One way to deflect some of the concerns associated with high fuel consumption of heavier-than-air aircraft is to rely on lighter-than-air vehicles. While these vehicles are often associated with spectacular accidents and slow motion, they also offer an unmatched capability to fly for long periods of time (more than 48 h) and to do so silently [44.24, 25]. Establishing control over lighter-than-air vehicles can be, however, somewhat challenging. In particular these vehicles are quite sensitive to winds and often tend to *go where the wind takes them*. Smaller platforms used for research must therefore evolve in closed environments [44.26]. The use of such vehicles for outdoor research can be quite daunting, because their large size requires considerable infrastructure to store them.

44.6 The Entry Level for Aerial Robotics: Inner-Loop Control

Aerial robots exhibit the complex flight dynamics associated with flight vehicles. As a consequence, precise motion control rapidly becomes a necessity for *any* aerial robotics activity to occur successfully. This means that effort must go into reliable basic flight control before more advanced, intelligent mission management can be attempted. Inner-loop control is achieved by the right sensing equipment, and by adequate control algorithms. Efficient hovering vehicles tend to be unstable, which makes their stabilization more difficult than that of purely cruising vehicles, for which many commercial control packages are now commonly available.

44.6.1 Sensing and Estimation

Airborne robots come with a variety of sensing options, which include

- inertial navigation systems (gyroscopes, accelerometers)
- global navigation satellite systems (GLONASS, GPS, Galileo)
- terrestrial radio navigation systems (VHF omnidirectional range (VOR), distance measuring equipment (DME), instrument landing system (ILS))
- air data probes and altimeters
- radar and passive vision sensors
- magnetic compasses
- distance measuring (altitude radars, ultrasonic sensors, and laser range finders)

The choice of sensors is critical to obtaining a properly flying robot. Usually, the same suite of sensors may not apply to all phases of flight. We will concentrate our discussion on the first four sensor types.

Inertial Navigation Systems

Inertial measurement systems consist of a combination of usually three orthogonally mounted accelerometers

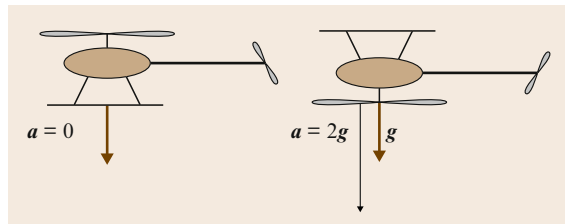


Fig. 44.10 Two inertially equivalent helicopter configurations

and three orthogonally mounted gyroscopes (more may be used for the purpose of achieving redundancy). The accelerometer suite measures, up to sensor error, the accelerations experienced by the vehicle at the location of the inertial sensor minus gravity. Gyroscopes measure vehicle angular velocities. Modern inertial sensors are usually rigidly linked to the vehicle to form strap-down inertial measurements systems. Such systems have become very cheap and very popular. Unlike many nonflying applications, unaided inertial measurement packages are not sufficient for estimating the attitude of an airborne vehicle. Indeed, consider Fig. 44.10, showing two helicopters equipped with inertial measurement systems. One, straight up, is hovering. The second, upside down, races towards the ground with an acceleration of $2g$. The accelerations and rotation rates recorded by the onboard inertial measurement unit will be strictly the same in both cases. While attitude can be estimated by integrating angular rates over time, the approach will eventually fail without correcting for accumulated error from another source. It remains that inertial measurement units are extremely useful to measure variations in acceleration and angular velocities, and constitute a staple of inner-loop control systems. Small radio-controlled helicopters now come with built-in gyroscopic yaw dampers that make their manual operation much more manageable. A key progress was made when Analog Devices introduced a low-cost micromechanical gyroscope [44.27]. Since then, this technology appears to have made its way into most commercially available inertial measurement units (IMUs), thereby greatly reducing their cost and weight. Practical inertial navigation systems on aircraft typically receive at least position updates from other sensors (discussed below), called inertial aiding.

Global Navigation Satellite Systems

The Global Positioning System (GPS) and its Russian equivalent GLONASS and future European Galileo space-based systems offer real-time absolute position information, using a constellation of satellites circumnavigating the earth. Ever since the beginning of their operation, global navigation systems (GNS) have been the object of a debate concerning their use in aerial robotics, with many researchers recommending against using such a large manmade navigation infrastructure to achieve *true autonomy*. Their arguments tend to become justified by the occurrence of recent needs in such applications as Mars exploration and low-altitude flight

in obstacle-laden environments (such as cities) where satellite-based navigation is often unavailable. Whenever they are available, however, satellite navigation systems are a convenient and cheap means for a vehicle to locate itself. This modest investment has often been the enabler of automatic flight for many researchers and is currently used by virtually all existing industrial systems. Pushed to their limits, satellite navigation systems have been shown to achieve the entire range of desired navigation and sensing functions, which include vehicle position and attitude: GPS-only flight for a small helicopter robot was achieved in 1995 at Stanford University [44.28].

Altimeter and Air Data Probes

Pressure-measuring devices are immensely useful sensors in aerial robotics. With ingenious arrangements of pressure sensors (such as pitot tubes) it is possible to measure (1) the atmospheric pressure at the location of the robot and (2) the so-called *dynamic pressure*, $\rho v^2/2$, along all vehicle axes. These data can themselves be transformed into precious information about the aerial robot's altitude and direction of motion relative to the air it is flying in. Depending on the vehicle used, air data probes may be challenging to build and constitute an interesting field of investigation. Indeed, pressure probes are very sensitive to flow perturbations generated by fuselage, wings, and most importantly rotors and propellers. Air data probes are therefore positioned as far away from the main elements of the vehicle as possible (for example, along a boom extending forward of the vehicle fuselage). Figure 44.11 shows one such air data probe configuration. Mounted together with inertial measurement systems, air data probes allow aircraft to maintain stable flight at a prescribed altitude. With the current state of technology, they remain somewhat insufficient to achieve, alone, stable hovering flight for helicopters.

Passive Vision

Passive vision has become a very popular sensor for inner-loop control. Even unsophisticated light sensors able to differentiate between the intensity of infrared activity from the ground versus that emitted by the sky has made its way into small commercial products, mostly aimed at assisting remote-controlled vehicle flight. As will be discussed later, passive vision devices have also found applications for vehicle–obstacle and vehicle–vehicle proximity management, and for landing applications. Recent research aimed at using vision for inner-loop control applications includes work aim-

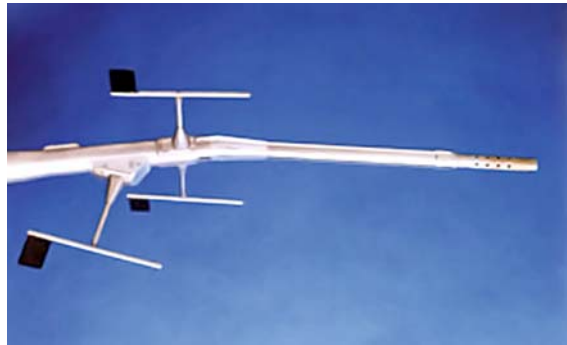


Fig. 44.11 Air data probe (Source: NASA Dryden Flight Research Center). The vanes are used to measure angle of attack and sideslip angle

ing at tracking relatively invariant features such as the horizon [44.29].

44.6.2 Estimator Design

The individual inputs collected from each sensor are usually not sufficient to estimate the state of the vehicle. Different sensors may be efficient over different flight regimes. The proper way to leverage individual information provided by each sensor is through an appropriate *filtering* process that can yield rather comprehensive information about the entire system's state. Unlike ground robots, the necessity for good robot state estimates arises early in the robot development process since closed-loop flight would be impossible otherwise. However, the structure of the filters is usually a great deal simpler than their ground-based equivalent, since the difficulty of flight is compensated by a rather simple and uniform environment structure. As a consequence, simple filters such as (extended) Kalman filters are usually enough for a large number of applications and quickly enable flight [44.30].

44.6.3 Inner-Loop Control

Inner-loop control of aerial vehicles naturally builds upon the previously discussed state estimator, and is relatively easy for routine flight operations. By this we mean that the process by which a good vehicle controller is obtained only requires following standard textbook techniques such as proportional, integral, and derivative control or linear quadratic control [44.31–33], appropriately scheduled against essential parameters such as vehicle speed and altitude. As a result, several research groups, and now several companies, have built basic

guidance and control packages suitable for aerial robots, both fixed-wing and helicopter.

Among the notable recent advances for the inner-loop control of aerial robots, we find the successful

experimental application of adaptive and learning control techniques [44.21, 34, 35], which offer stable controlled helicopter flight from very coarse initial vehicle dynamics knowledge.

44.7 Active Research Areas

This section presents some of the active research in aerial robotics. Such research efforts aim at answering the challenges outlined in Sect. 44.4.

44.7.1 Interfacing with the Human Infrastructure: Meeting the Regulations

While manned flight operations indeed have an excellent safety record, the price paid for this safety is a strong specialization of those regulations to human-operated systems, and a slow evolution of these regulations towards accepting aerial robots of all sizes. Currently none of the existing aerial robots is able to meet these regulations in the absence of a human pilot. This includes the ability to see (or sense) and avoid other aircraft, comply with air-traffic rules, and operate harmoniously with the current ground-based, manned air-traffic control system [44.4, 5, 36]. In the case of the US however, the FAA has acknowledged the economic potential of aerial robots by opening an office specifically devoted to such systems (the Unmanned Aircraft System Group), and by delivering permits to fly over certain areas (especially disaster areas) within a couple of hours of a request. However, the FAA currently emphasizes access for remotely piloted machines (as opposed to fully autonomous machines), where a ground-based pilot must have the means to communicate by voice with the FAA control center in charge of the geographical area where the vehicle is operated.

Efforts to help aerial robots improve their interface with other vehicles include the adaptation of existing systems to prevent mid-air collisions between aerial robots and other traffic, the development of *see-and-avoid procedures*, and means to interact with an aerial robot as one would interact with a human pilot (natural language interfaces). We now detail these efforts.

Collision Avoidance for Remotely Piloted Vehicles

The most immediate efforts aimed at inserting aerial robots in the civilian airspace consists of adapting existing airborne collision avoidance systems (ACAS),

originally designed for manned systems, to aerial robots. Such systems are based on cooperative position information sharing between aircraft extracted from radar-based navigation and surveillance systems. The reason for emphasizing such systems over other, newer technologies is that they have already undergone extensive, and expensive, development, validation and testing. As such, several unmanned vehicles, such as the Global Hawk unmanned aircraft, are now fitted with ACAS systems [44.37]. However, such systems are not automated, meaning that the remote human pilot ultimately decides whether to execute the maneuvers recommended by the system. The possibility of completely automating such collision avoidance systems is the object of recent studies [44.38], with the clear intent of fitting them on aerial robots. However, the weight of ACAS system hardware, as well as the power required to operate them, makes such systems suitable only for large vehicles. In the context of rapidly evolving technology (such as the generalization of positioning information using global navigation satellite systems), current ACAS technology may also become rapidly obsolete.

Sense and Avoid

The idea, encouraged by institutional service providers, is to ensure that aerial robots are able to detect the presence of other traffic and avoid it as necessary and at least as well as a human pilot. Several candidate sensing technologies are currently in development, including passive vision systems [44.39–42], in an effort aimed at making aerial robots as able as humans to avoid other traffic when the sky is clear (visual flight rules). While much can be done in the visible spectrum, concerns over vehicle flight in clouds raises the necessity to consider other frequency bands, such as the near infrared, if aerial robots will need to be able to detect other traffic better than a human pilot [44.43].

Human Interfacing

Another active research venue is to facilitate the interfacing of robots with humans (e.g., an aerial robot interacting with a human air-traffic controller). Recent research in human–aerial robot interaction has

shown that aerial robots can interact productively with humans, by combining natural language processing interfaces with advanced vehicle path and task planning capabilities [44.44–46]. A natural interface might capture standard, unambiguous phraseology such as North Atlantic Treaty Organization (NATO) phraseology or air-traffic control phraseology. The impact of such technology on aerial robots would be profound since they would then be able to enter airspace with little or no visibility and be able to interact with the predominantly ground-based, human-intensive air-traffic control structure.

44.7.2 High-Agility Flight

One of the important characteristics of aerial robots is the ability to operate at the limit of its structural strength, unimpeded by the presence and physiological limitations of a human pilot. This allows aerial robots, especially small ones, to operate very aggressively. As a result, several research groups have explored the possibility of achieving aggressive flight with either fixed-wing or rotary-wing vehicles. The key factors that have enabled the onset of highly aggressive flight has been the emergence of lightweight computing environments and sensors, notably GPS and inertial systems. Indeed, aggressive flight (where aggressive flight refers to any abrupt change in vehicle attitude) is closely related to available vehicle mass and size, as discussed earlier.

Figure 44.12 shows the evolution of three helicopter configurations over time. While the vehicle platform has evolved little or not at all, the onboard avionics has progressively shrunk. In the mid-1990s, the onboard avionics typically would weigh the same or more than the helicopter mass. By the early 2000s, the onboard avionics would be about half of the vehicle mass, while by the mid-2000s the onboard avionics represent only a small fraction of the helicopter mass.

The corresponding levels of achievable agility have evolved correspondingly. By the early 2000s, basic aerobatic maneuvers became feasible [44.47], and by 2007 fully fledged aerobatics had been reported [44.48]. Other efforts involving unusual flight attitudes and fault recovery include those of Chiba University (Japan), who demonstrated autorotation landings for autonomous helicopters [44.49].

Parallel to rotorcraft agile flight, several efforts have also successfully enabled aerial agility for fixed-wing robots [44.21].

44.7.3 Take-Off, Landing, and Interaction with Other Vehicles

One of the richest current areas of investigation for aerial robots involves vehicle operation next to other vehicles or infrastructures. These operations include take-off, landing, docking, and separation.

Take-Off and Landing

Take-off and landing experimentation and research is proving particularly interesting for small-sized aerial robots. Indeed, the dynamics of smaller vehicles enable strong departures from conventional, manned-vehicle take-off and landing operations, for example, most fixed-wing unmanned aerial vehicles under 5 kg are better off simply flying into the ground than attempting to land in a smooth fashion. One of the best illustrations of how vehicle landing procedures may dramatically change for smaller-sized aerial robots is Insitu's and Hood Technology's *Skyhook* concept: small, fixed-wing aerial robots are recovered by allowing them to catch a vertical cable with the tip of one of their wings [44.12]. The cable itself is held by means of a crane, itself mounted on a surface vehicle (e.g., truck or ship). At take-off, similar scaling considerations apply, with many fixed-wing vehicles being launched by hand or by means of a catapult. One of the consequences of the increased tolerance of small ve-



Fig. 44.12a–c Avionics versus vehicle. (a) Stanford Helicopter (c. 1995). (b) MIT helicopter (c. 2001). (c) Stanford Helicopter (c. 2006)



Fig. 44.13 Skyhook system in action (courtesy Insitu, Inc.)

hicles to *crash landings* is also their reduced need for high-resolution navigation information, for example, it has been shown possible to land small-sized vehicles on a designated target with monocular vision only [44.50].

Helicopter robots have, so far, not benefitted from the same kind of developments, and much of their take-off and landing procedures are similar to their larger counterparts. The main reason may be attributed to the presence of a fragile rotor that spins at high speed, and that must avoid contact with other vehicles or the ground. Many of the current robotic helicopter landing procedures simply consist of hovering above the landing area, then commanding a limited descent rate until the vehicle records it has touched the ground. More

challenging situations (e.g., sloped terrain or moving platforms), traditionally handled by humans in large platforms, remain difficult for aerial robots. For this reason, the *helicopter landing problem* has attracted the attention of many research teams. On the one hand, there have been many efforts combining advanced sensing environments [44.51–55] with advanced control algorithms to enable affordable landing in structured environments which are not simply horizontal landing pads. On the other hand, identifying suitable landing places in unprepared environments by means of remote sensing and signal processing is also an area of active research [44.56–58].

Operations in the Vicinity of Other Vehicles: Docking and Undocking

Docking operations for unmanned aerial vehicles are necessary to improve their range and autonomy. Indeed, it is conceivable that some optimal aerial robot configuration consist of a parent–child system, whereby a larger machine provides a primary deployment and retrieval mechanism for several smaller vehicles. Such a concept has existed for a long time, with airships acting as carriers for smaller aircraft [44.59]. More recently however, it is in-flight aerial refueling that has motivated recent research on docking aerial robots. Indeed, the possibility for such vehicles to refuel considerably increases their operational range [44.60–62]. The NASA Dryden Flight Research Center has recently reported the completion of the first vision-aided fully automated aerial refueling operation, using computer vision for the purpose of recognizing and tracking the fuel hose



Fig. 44.14a,b Automatic airborne refueling. (a) Typical refueling configuration. (b) Camera view of refueling basket (Source: NASA Dryden Flight Research Center)



Fig. 44.15 Georgia Tech's unmanned helicopter in a parent–child configuration. The child is a hover-capable ducted fan

that must be captured by the aerial robot, as shown in Fig. 44.14.

Undocking operations are comparatively easier to perform. They remain, however, spectacular since the dynamics of the aerial robot dramatically change as it is dropped from its mother ship. An extreme example of such a situation is illustrated by Georgia Tech's successful dropping of a small ducted fan aerial robot from a larger autonomous helicopter. The small ducted fan then successfully stabilized itself. Pictures of this experiment are shown in Fig. 44.15.

44.7.4 Reactive Flight in Cluttered Environments and Obstacle Avoidance

Flight in cluttered environments includes any phase of the flight where vehicles are in close proximity to obstacles. This flight mode is particularly important for low-altitude applications. Several achievements have been reported in this area in the recent past, using a variety of sensing techniques.

Among the first significant works relying on passive vision techniques, *Beard* and *McLain*'s certainly stands out as one of the most entertaining and spectacular [44.63], using fixed-wing vehicles performing autonomous flight within a canyon using low-cost, optical flow computation techniques.

Other institutions involved with active as well as passive sensing techniques for vehicle navigation in cluttered environments and obstacle avoidance include Carnegie-Mellon University [44.64], where the authors report fast vehicle flight in highly cluttered environments, including obstacles as difficult to deal with as suspended cables. The NASA Ames research center also recently reported successes along similar lines as part of their work on adaptive landing in unprepared environments [44.56–58].

44.7.5 Path Planning and Higher-Level Planning Capabilities

Single-Robot Path Planning

Path planning for aerial robots resembles path planning for any robot, with the following distinctive characteristics: aerial robots are able to fly very fast (or may *have to* fly fast). Thus there is the distinct possibility of significant discrepancies between intended and actual trajectories. The vehicle dynamics must be fully accounted for when designing trajectories. Several path planning concepts have been proposed to handle this

problem, including [44.65–67] and many others. Another key issue in aerial robot trajectory planning arises when there is a discrepancy between the complexity of the environment and the maneuvering space needed for the vehicle. When planned for finite time or geographical horizons, it becomes important that a planner constantly keep a feasible loitering solution within the known environment [44.46].

Multirobot Path Planning and Coordination

There has recently been a surge in research activities for multivehicle path planning and coordination. Such research activities have been motivated by problems as diverse as the generation of noncolliding paths, the generation of *swarming* behaviors for applications such as phased-array, robot-borne antenna systems, collaborative target detection and prosecution, and collaborative search for thermal currents.

This rich literature, of which only a few references have been cited, stems from the conjunction of several constraints in the problem under study, including

- highly constrained dynamical systems (with restricted radius of curvatures and minimum speed requirements, for example)
- a variety of information management possibilities (including centralized, decentralized, distributed information)
- catastrophic consequences in case of failures

Initial work aimed at studying aerial-robot coordination from the perspective of mission execution include [44.68–71]. Swarming behaviors, or the ability for a vehicle group to generate a coherent, consensual behavior using only local information, has become the focus of much attention in the research community since the recent paper [44.72].

Collision avoidance has also formed the motivation for much research in multirobot coordination, see for example [44.73–75].

44.7.6 Integrated Aerial Robotic Operations: Aerial Robotics Contests

Research on the ability of aerial robots to perform completely autonomous missions, especially at low altitudes, is clearly well represented in contests such as the International Aerial Robotics Competition, initiated in 1991 by *Michelson* [44.1]. In this contest, universities, possibly supported by industry and government, compete against each other by demonstrating how their vehicles,

or vehicle systems, meet the requirements of the competition. A basic tenet of the competition is that the small aerial robotic systems entrants must be capable of complete autonomy (no human interaction) during the mission.

The rules of the competition have evolved from the inception of this effort to reflect advances in the capabilities of the proposed systems. One of the key characteristics of the competition is that it has always emphasized the simultaneous demonstration of several robotic functionalities, including basic mission execution, object reconnaissance and detection, and object manipulation. During the early days of the competition, the task asked for an aerial robot to recognize and pick up an object in a designated area, and carry it to another designated area. As universities were able to meet the initial challenges posed by the competition rules, the rules have evolved to a higher level of sophistication. As of today, the competition rules require complete autonomous operation of the vehicles over longer distances. The vehicles must now find and reach a village from a distance of three kilometers. They must also evolve towards higher reasoning capabilities about the objects and events being encountered. Moreover, emphasis has been placed on multimodal robotics, since the robotic system must be able to enter a building and explore it, a task currently best performed by ground robots.

Recognizing the growing gap between experienced participants and new entrants, several different competition levels have been established. While US participation in the competition is predominant, several non-US participants are also present, including Germany, England, Switzerland, Canada, and India. In 2000, the Technische Universitaet Berlin won the contest. Other contest win-

ners include Carnegie-Mellon University, the Georgia Institute of Technology, MIT/Draper Laboratory, and Stanford University.

Other aerial robotic competitions have since been established. For example, the French governmental organization Delegation Generale pour L'Armement (DGA), together with the Supaero and Ecole Nationale Supérieure Des Constructions Aeronautiques (ENSICA) engineering schools have proposed a contest involving very small-sized aerial robots in 2004, with a focus on their flight mechanics at various flight regimes. The government of Queensland, Australia together with the local research organizations Commonwealth Scientific and Industrial Research Organization (CSIRO) and Queensland University of Technology launched a new contest focusing on search-and-rescue missions in 2007.

The distribution of vehicle types involved in these contests is very different from the distribution of operational aerial robots. While operational aerial robotic systems are overwhelmingly of fixed-wing type, the machines used by universities during these contests offer a much more balanced distribution of aircraft and rotorcraft. Several reasons contribute to these differences and they have been outlined earlier. The operation of fixed-wing aircraft at relatively high altitude, for reconnaissance and surveillance missions offers a large and technologically easy market to reach, although it faces significant regulatory constraints. In comparison, the operation of small vehicles in cluttered environments definitely favors hovering-like vehicles. However, these vehicles, like other robots, face significantly more constraints in terms of environment sensing, obstacle avoidance, and task planning and execution complexity. As such, they are closer to the realm of basic research typical of universities.

44.8 Conclusions and Further Reading

Aerial robots represent a very interesting and exciting area of robotics, involving very dynamic platforms whose size ranges from a few centimeters to several tens of meters. It seems highly probable they will continue to see new applications, beginning with those that happen in relatively unpopulated areas and relatively high altitudes. The current applications of aerial robots are focused primarily on military operations. However, an ambitious civilian market led by Japan is currently burgeoning.

Aerial robots currently pose a challenge to all regulatory agencies, which must find modalities and rules

to insert them into airspace occupied by other traffic such as manned systems. The resulting technical research challenges include the development of a proper and affordable *sense-and-avoid* technology, and the ability for aerial robots to be conversant with other traffic and the ground control infrastructure.

Lower-altitude aerial robotics, often operating in cluttered environments, offers the opportunity to explore many generic robotics topics, including vision, path planning, mapping, and other algorithms in a progressive manner, while offering potential benefit of still more important applications.

A very dynamic research and development field, aerial robotics can be seen from a historical perspective by reading [44.3]. A snapshot of current UAV technology can be obtained, for example, from [44.76]. The lack of a known comprehensive, book-like presentation

of aerial robotics and its applications clearly indicates that the field is still very young, that operational experience is slowly building up, and that many challenges, most notably regulatory and safety challenges, must still be overcome.

References

- 44.1 R.C. Michelson: International aerial robotics competition – The world's smallest intelligent flying machines, 13th RPVs/UAVs International Conference, UK (1998) pp. 31.1–31.9
- 44.2 J.D. Anderson: Samuel Pierpont Langley – America's first aeronautical engineer, 38th AIAA Aerospace Sciences Meeting and Exhibit, Reno (2000) paper 0164
- 44.3 L.R. Newcome: *Unmanned Aviation, a Brief History of Unmanned Aerial Vehicles* (American Institute of Aeronautics and Astronautics, Reston 2004)
- 44.4 R.E. Weibel, R.J. Hansman: Safety considerations for operation of different classes of UAVs in the national airspace system, AIAA 3rd Unmanned Unlimited Technical Conference, Chicago (2004) paper 6244
- 44.5 R.E. Weibel, R.J. Hansman: An Integrated Approach to Evaluating Risk Mitigation Measures for UAV Operational Concepts in the National Airspace System, AIAA Infotechs., Aerospace, Arlington (2005) paper 6957
- 44.6 D. Hughes: UAVs face hurdles in gaining access to civil airspace, *Aviation week* (2007)
- 44.7 K.C. Wong, C. Bil, G. Gordon, P.W. Gibbens: Study of the Unmanned Aerial Vehicle (UAV) Market in Australia, Aerospace Technology Forum Report (1997)
- 44.8 R. Sugiura, N. Noguchi, K. Ishii, H. Terao: Development of remote sensing system using an unmanned helicopter, *J. JSAM* **65**(1) (2003)
- 44.9 R. Sugiura, T. Fukagawa, N. Noguchi, K. Ishii, Y. Shibata, K. Toriyama: Field information system using an agricultural helicopter towards precision farming, IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Kobe 2003) pp. 1073–1078
- 44.10 K. Schutte, H. Sahli, D. Schrottmayer, F.J. Varas: ARC: A camcorder based mine field detection system, Fifth International Airborne Remote Sensing Conference, San Francisco (2001)
- 44.11 S.E. Wright: UAVs in Community Police Work, AIAA Infotechs., Aerospace, Arlington (2005) paper 6955
- 44.12 T. McGeer, S. Sliwa, D. Liwa: Commercial applications for UAVs within the fishing industries, AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles, Portsmouth (2002) paper 3401
- 44.13 V. Ramanathan: Maldives AUAV Campaign (MAC): Observing Aerosol–Cloud–Radiation–Climate Interactions Simultaneously from Three Stacked Autonomous Unmanned Aerial Vehicles (AUAVs), Tech. Rep., Center for Atmospheric Sciences and the Center for Clouds, Chemistry and Climate Scripps Institution of Oceanography, University of California, San Diego (2006)
- 44.14 M. Patterson, A. Mulligan, J. Douglas, J. Robinson, J. Pallister: Volcano Surveillance by ACR Silver Fox, AIAA Infotechs., Aerospace, Arlington (2005) paper 6954
- 44.15 North Atlantic Treaty Organization: Standard Interfaces of UAV Control System (UCS) for NATO UAV Interoperability, NATO STANAG, 4586 (2004)
- 44.16 T. Jordan, J. Foster, R. Bailey, C.M. Belcastro: AirSTAR: A UAV platform for flight dynamics and control system testing, 25th AIAA Aerodynamic Measurement Technology and Ground Testing Conference, San Francisco (2006) paper 3307
- 44.17 H. Tennekes: *The Simple Science of Flight: From Insects to Jumbo Jets* (MIT Press, Cambridge 1992–97)
- 44.18 J.D. Anderson: *Introduction to flight* (McGraw-Hill, New York 1989)
- 44.19 R.C. Nelson: *Flight Stability and Automatic Control* (McGraw Hill, New York 1998)
- 44.20 R.D. Braun, H.S. Wright, M.A. Croom, J.S. Levine, D.A. Spencer: The Mars airplane: A credible science platform, IEEE Aerospace Conference (Big Sky 2004) p. 408
- 44.21 A.D. Wu, M.A. Turbe, S. Kannan, J.C. Neidhoefer, E.N. Johnson: Flight Results of Autonomous Airplane Transitions to and from Vertical Hover, AIAA Guidance, Navigation and Control Conference, Keystone (2006) paper 6775
- 44.22 W.R. Williamson, M.F. Abdel-Hafez, I. Rhee, J.D. Wolfe, J.L. Speyer: A Formation Flight Experiment Using Differential Carrier Phase for Precise Relative Navigation, Institute of Navigation – GPS meeting (2002)
- 44.23 J. Sanderson: *Private Pilot Manual* (Jeppesen Sanderson, Englewood 1984)
- 44.24 I. Schafer, R. Kuke, P. Lindstrand: Airships as Unmanned Platforms: Challenge and Chance, AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles, Portsmouth (2002) paper 3423
- 44.25 N. Mayer: Lighter-than-air systems, *Aerosp. Am.* **12**, 30–31 (2005)

- 44.26 E.A. Olsen, C.-W. Park, J.P. How: 3D Formation Flight using Differential Carrier-phase GPS Sensors, Institute of Navigation GPS Meeting, Nashville (1998)
- 44.27 J.A. Geen, S.J. Sherman, J.F. Chang, S.R. Lewis: Single-chip surface micromachined integrated gyroscope with 50/spl deg/h Allan deviation, IEEE J. Solid-State Circ. **37**(12), 1860–1866 (2002)
- 44.28 A. Conway: Autonomous Control of an unstable model helicopter using carrier-phase GPS only. Ph.D. Thesis (Stanford University, Stanford 1995)
- 44.29 T.P. Webb, J. Prazenica, A.J. Kurdila, R. Lind: Vision-Based State Estimation for Autonomous Micro Air Vehicles, AIAA Guidance, Navigation and Control Conference (San Francisco 2005) paper 5869
- 44.30 B. Anderson, J.B. Moore: *Optimal Filtering* (Prentice-Hall, Upper Saddle River 1979)
- 44.31 A.E. Bryson, Y.C. Ho: *Applied Optimal Control* (Hemisphere, Washington 1975)
- 44.32 A.E. Bryson: *Control of Spacecraft and Aircraft* (Princeton Univ. Press, Princeton 1994)
- 44.33 G.F. Franklin, J.D. Powell, A. Emami-Naeni: *Feed-back Control of Dynamic Systems* (Addison-Wesley, Reading 1986)
- 44.34 E.N. Johnson, A.J. Calise: Limited Authority Adaptive Flight Control for Reusable Launch Vehicles, AIAA J. Guidance Control Dyn. **26**(6), 906–913 (2003)
- 44.35 A.Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, E. Liang: Inverted autonomous helicopter flight via reinforcement learning. In: *Experimental Robotics*, ed. by M. Ang, O. Khatib (Springer, Berlin, Heidelberg 2006)
- 44.36 R. Clothier, R. Walker: Determination and Evaluation of UAV Safety Objectives, 21st International Unmanned Air Vehicle Systems Conference, Bristol (2006) pp. 18.1–18.16
- 44.37 J. Kuchar, J. Andrews, A. Drumm, T. Hall, V. Heinz, S. Thompson, J. Welch: A Safety Analysis Process for the Traffic Alert and Collision Avoidance System (TCAS) and See-and-Avoid Systems on Remotely Piloted Vehicles, AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit, Chicago (2004) paper 6423
- 44.38 J.L. Less, J. Chen: Flight testing the automatic air collision avoidance system (AUTO ACAS), Adv. Astronaut. Sci. **118**, 133–144 (2004)
- 44.39 R.J. Carnie, R.A. Walker, P.I. Corke: Computer Vision Based Collision Avoidance for UAVs, 11th Australian International Aerospace Congress, Melbourne (2005)
- 44.40 R. Carnie, R. Walker, P. Corke: Image processing algorithms for UAV sense and avoid, IEEE International Conference on Robotics and Automation Orlando, Melbourne (2006) pp. 2848–2853
- 44.41 J. McCalmont, J. Utt, M. Deschenes: Detect and avoid technology demonstration, 1st Unmanned Aerospace Vehicles, Systems, Technologies and Operations Conference and Workshop, Portsmouth (2002)
- 44.42 J. Utt, J. McCalmont, M. Deschenes: Test and Integration of a Detect and Avoid System, AIAA Third Unmanned Unlimited Technical Conference, Workshop and Exhibit, Portsmouth (2004) paper 6424
- 44.43 J.F. McCalmont, J. Utt, M. Deschenes, M.J. Taylor: Sense and avoid technology for Global Hawk and Predator UAVs, Infrared Technology and Applications, XXXI SPIE Conference (2005) pp. 684–692
- 44.44 P. Doherty, G. Granlund, K. Kuchcinski, E. Sandewall, K. Nordberg, E. Skarman, J. Wiklund: The WITAS Unmanned Aerial Vehicle Project, 14th European Conference on Artificial Intelligence (Berlin 2000) pp. 747–755
- 44.45 E. Sandewall, P. Doherty, O. Lemon, S. Peters: Words at the right time: Real-time dialogues with the WITAS unmanned aerial vehicle. In: *KI 2003: Advances in Artificial Intelligence*, ed. by A. Gunter, R. Kruse, B. Neumann (Springer Lectur Notes in Artificial Intelligence, Berlin 2003) pp. 52–78
- 44.46 T. Schouwenaars, M. Valenti, E. Feron, J. How, E. Roche: Linear programming and language processing for human/unmanned-aerial-vehicle team missions, AIAA J. Guid. Control Dyn. **29**(2), 303–313 (2006)
- 44.47 V. Gavrillets, B. Mettler, E. Feron: Human-inspired control logic for automated maneuvering of a miniature helicopter, AIAA J. Guid. Control Dyn. **27**(5), 752–759 (2004)
- 44.48 P. Abbeel, A. Coates, M. Quigley, A.Y. Ng: An application of reinforcement learning to aerobatic helicopter flight., Neural Information Processing Systems Conference, Vancouver (2006)
- 44.49 K. Nonami: Personal communication (2003)
- 44.50 A. Proctor, E. Johnson: Vision-Only Approach and Landing, AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco (2005) paper 5871
- 44.51 P. Corke: An inertial and visual sensing system for a small autonomous helicopter, J. Robot. Syst. **21**, 43–51 (2004)
- 44.52 S. Saripalli, G.S. Sukhatme: Landing on a moving target using an autonomous helicopter, International Conference on Field and Service Robotics, Mt Fuji (2003)
- 44.53 S.S. Sastry, C.S. Sharp, O. Shakernia: A vision system for landing an unmanned aerial vehicle, IEEE International Conference on Robotics and Automation (Seoul 2001) pp. 1720–1727
- 44.54 G. Tournier, M. Valenti, J. How, E. Feron: Estimation and Control of a quadrotor vehicle using monocular vision and moiré patterns, AIAA Guidance, Navigation and Control, Keystone (2006) paper 6711
- 44.55 T. Merz, S. Duranti, G. Conte: Autonomous landing of an unmanned helicopter based on vision and inertial sensing. In: *Experimental Robotics*, ed.

- by M. Ang, O. Khatib (Springer, Berlin, Heidelberg 2006)
- 44.56 C. Theodore, S. Sheldon, D. Rowley, T. McLain, W. Dai, M. Takahashi: Full mission simulation of a rotorcraft unmanned aerial vehicle for landing in a non-cooperative environment, 61st Annual Forum of the American Helicopter Society, Grapevine (2005)
- 44.57 C. Theodore, D. Rowley, D. Hubbard, A. Ansar, L. Matthies, S. Goldberg, M. Whalley: Flight trials of a rotorcraft unmanned aerial vehicle landing autonomously at unprepared sites, American Helicopter Society 62nd Annual Forum, Phoenix (2006)
- 44.58 C. Theodore, M. Tischler: Precision autonomous landing adaptive control experiment (PALACE), 25th Army Science Conference, Orlando (2006)
- 44.59 D. Topping: *When Giants Roamed the Sky: Karl Arnstein and the Rise of Airships from Zeppelin to Goodyear* (Univ. of Akron Press, Akron 2001)
- 44.60 L. Pollini, G. Campa, F. Giulietti, M. Innocenti: Virtual Simulation Set-Up for UAVs Aerial Refuelling, AIAA Modeling and Simulation Technologies Conference and Exhibit, Austin (2003) paper 5682
- 44.61 J. Valasek, J. Kimmet, D. Hughes, K. Gunnam, J. Junkins: Vision based sensor and navigation system for autonomous aerial refueling, AIAA's 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles, Portsmouth (2002) paper 3441
- 44.62 J. Hansen, J. Murray, N. Campos: The NASA Dryden AAR Project: A Flight Test Approach to an Aerial Refueling System, AIAA Atmospheric Flight Mechanics Conference and Exhibit, Providence (2004) paper 4939
- 44.63 J.B. Saunders, B. Call, A. Curtis, R.W. Beard, T.W. McLain: Static and dynamic obstacle avoidance in miniature air vehicles, AIAA Infotechs., Aerospace, Arlington (2005) paper 6950
- 44.64 S. Scherer, S. Singh, L. Chamberlain, S. Saripalli: Flying fast and low among obstacles, International Conference on Robotics and Automation, Roma (2007) pp. 2023–2029
- 44.65 E. Frazzoli, M. Dahleh, E. Feron: Maneuver-based motion planning for nonlinear systems with symmetries, IEEE Trans. Robot. **21**(6), 1077–1091 (2005)
- 44.66 R. Murray, Z. Li, S. Sastry: *A Mathematical Introduction to Robotic Manipulation* (CRC, Boca Raton 1993)
- 44.67 A. Richards, J. How, T. Schouwenaars, E. Feron: Spacecraft trajectory planning with collision and plume avoidance, AIAA J. Guid. Control Dyn. **25**(4), 755–764 (2002)
- 44.68 P.R. Chandler, M. Pachter, S. Rasmussen: UAV cooperative control, American Control Conference, Arlington (2001) pp. 50–55
- 44.69 M. Alighanbary, Y. Kuwata, J.P. How: Coordination and Control of Multiple UAVs with Timing Constraints and Loitering, American Control Conference (2003) pp. 5311–5316
- 44.70 T. Lemaire, R. Alami, S. Lacroix: A distributed tasks allocation scheme in multi-UAV context, IEEE International Conference on Robotics and Automation, New Orleans (2004) pp. 3622–3627
- 44.71 Y. Jin, A.A. Minai, M.M. Polycarpou: Cooperative real-time search and task allocation in UAV teams, IEEE Conference on Decision and Control, Maui (2003) pp. 7–12
- 44.72 A. Jadbabaie, J. Lin, A.S. Morse: Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Trans. Automat. Control **1**(4), 221–231 (2003)
- 44.73 L. Pallottino, A. Bicchi, E. Frazzoli: Probabilistic verification of decentralized multi-agent control strategies: a Case Study in Conflict Avoidance, American Control Conference, New-York (2007) pp. 170–175
- 44.74 L. Pallottino, A. Bicchi: On optimal cooperative conflict resolution for air traffic management systems, IEEE Trans. Intell. Transport. Syst. **1**(4), 221–231 (2000)
- 44.75 C.J. Tomlin, I. Mitchell, A. Bayen, M. Oishi: Computational techniques for the verification of hybrid systems, Proc. IEEE **91**(7), 986–1001 (2003)
- 44.76 J.R. Wilson: UAV Worldwide Roundup 2007, Aerosp. Am. **5**, 30–37 (2007)

Space Robots

45. Space Robots and Systems

Kazuya Yoshida, Brian Wilcox

In the space community, any unmanned spacecraft can be called a robotic spacecraft. However, *space robots* are considered to be more capable devices that can facilitate manipulation, assembling, or servicing functions in orbit as assistants to astronauts, or to extend the areas and abilities of exploration on remote planets as surrogates for human explorers.

In this chapter, a concise digest of the historical overview and technical advances of two distinct types of space robotic systems, orbital robots and surface robots, is provided. In particular, Sect. 45.1 describes orbital robots, and Sect. 45.2 describes surface robots. In Sect. 45.3, the mathematical modeling of the dynamics and control using reference equations are discussed. Finally, advanced topics for future space exploration missions are addressed in Sect. 45.4.

Key issues in space robots and systems are characterized as follows. *Manipulation* – Although manipulation is a basic technology in robotics, microgravity in the orbital environment requires special attention to the motion dynamics of manipulator arms and objects being handled. Reaction dynamics that affect the base body, impact dynamics when the robotic hand contacts an object to be handled, and vibration dynamics due to structural flexibility are included in this issue. *Mobility* – The ability to locomote is particularly important in exploration robots (rovers) that travel on the surface of a remote planet. These surfaces are natural and rough, and thus challenging to traverse. Sensing and perception, traction mechanics, and vehicle dynamics, control, and navigation are all mobile robotics technologies that must be demonstrated in a natural untouched environment. *Teleoperation and autonomy* – There is a significant time delay between a robotic system at a work site and a human operator in an operation room on the Earth. In earlier orbital robotics demonstrations, the latency was typically 5 s, but

can be several tens of minutes, or even hours for planetary missions. Telerobotics technology is therefore an indispensable ingredient in space robotics, and the introduction of autonomy is a reasonable consequence. *Extreme environments* – In addition to the microgravity environment that affects the manipulator dynamics or the natural and rough terrain that affects surface mobility, there are a number of issues related to extreme space environments that are challenging and must be solved in order to enable practical engineering applications. Such issues include extremely high or low temperatures, high vacuum or high pressure, corrosive atmospheres, ionizing radiation, and very fine dust.

45.1 Historical Developments and Advances of Orbital Robotic Systems	1032
45.1.1 Space Shuttle Remote Manipulator System	1032
45.1.2 ISS-Mounted Manipulator Systems	1033
45.1.3 ROTEX	1035
45.1.4 ETS-VII	1035
45.1.5 Ranger	1036
45.1.6 Orbital Express	1036
45.2 Historical Developments and Advances of Surface Robotic Systems	1037
45.2.1 Teleoperated Rovers	1037
45.2.2 Autonomous Rovers	1039
45.2.3 Research Systems	1042
45.2.4 Sensing and Perception	1043
45.2.5 Estimation	1043
45.2.6 Manipulators for In Situ Science	1044
45.3 Mathematical Modeling	1044
45.3.1 Space Robot as an Articulated-Body System	1044
45.3.2 Equations for Free-Floating Manipulator Systems	1045
45.3.3 Generalized Jacobian and Inertia Matrices	1045
45.3.4 Linear and Angular Momenta	1047
45.3.5 Virtual Manipulator	1047

45.3.6	Dynamic Singularity.....	1047	45.4	Future Directions of Orbital and Surface Robotic Systems.....	1056
45.3.7	Reaction Null Space (RNS)	1048	45.4.1	Robotic Maintenance and Service Missions.....	1056
45.3.8	Equations for Flexible-Based Manipulator Systems	1048	45.4.2	Robonaut.....	1057
45.3.9	Advanced Control for Flexible Structure Based Manipulators	1049	45.4.3	Aerial Platforms	1057
45.3.10	Contact Dynamics and Impedance Control.....	1050	45.4.4	Subsurface Platforms	1059
45.3.11	Dynamics of Mobile Robots.....	1052	45.5	Conclusions and Further Reading.....	1060
45.3.12	Wheel Traction Mechanics	1053	References	1060	

45.1 Historical Developments and Advances of Orbital Robotic Systems

The first robotic manipulator arm used in the orbital environment was the Space Shuttle remote manipulator system. It was successfully demonstrated in the STS-2 mission in 1981 and is still operational today. This success opened a new era of orbital robotics and inspired a number of mission concepts to the research community. One ultimate goal that has been discussed intensively after the early 1980s is the application to the rescue and servicing of malfunctioning spacecraft by a robotic free-flyer or free-flying space robot (see, for example, the Space Application of Automation, Robotics and Machine Intelligence (ARAMIS) report [45.1], Fig. 45.1). In later years, manned service missions were conducted for the capture–repair–deploy procedure of malfunctioning satellites (Anik-B, Intelsat 6, for example) and for the maintenance of the Hubble space telescope (STS-61, 82, 103, and 109).

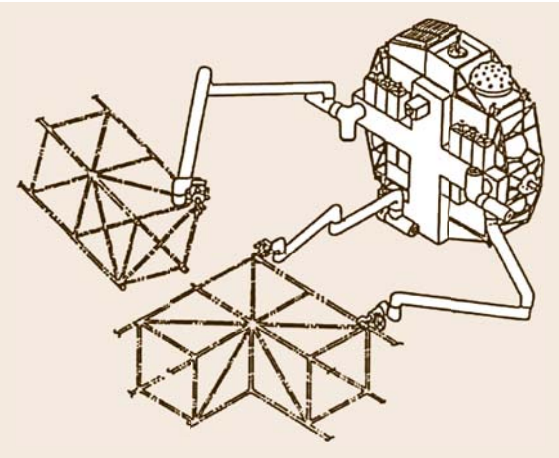


Fig. 45.1 A conceptual design of a telerobotic servicer [45.1]

For all of the above examples, the Space Shuttle, a manned spacecraft with dedicated maneuverability was used. However, unmanned servicing missions have not yet become operational. Although there were several demonstration flights, such as engineering test satellite (ETS)-VII and Orbital Express (to be elaborated later), the practical technologies for unmanned satellite servicing missions await solutions to future challenges.

45.1.1 Space Shuttle Remote Manipulator System

Onboard the Space Shuttle, the *shuttle remote manipulator system (SRMS)*, or Canadarm, is a mechanical arm that maneuvers a payload from the payload bay of the Space Shuttle orbiter to its deployment position and then releases it [45.2]. It can also grapple a free-flying payload and berth it to the payload bay of the orbiter. The SRMS was first used on the second Space Shuttle mission STS-2, launched in 1981. Since then, it has been used more than 100 times during Space Shuttle flight missions, performing such payload deployment or berthing as well as assisting human *extravehicular activities (EVAs)*. Servicing and maintenance missions to the Hubble space telescope and construction tasks of the International Space Station have also been successfully carried out by the cooperative use of the SRMS with human EVAs.

As depicted in Fig. 45.2, the SRMS arm is 15 m long and has six degrees of freedom, comprising shoulder yaw and pitch joints, an elbow pitch joint and wrist pitch, yaw, and roll joints. Attached to the end of the arm is a special gripper system called the *standard end-effector (SEE)*, which is designed to grasp a pole-like fixture (GF) attached to the payload.

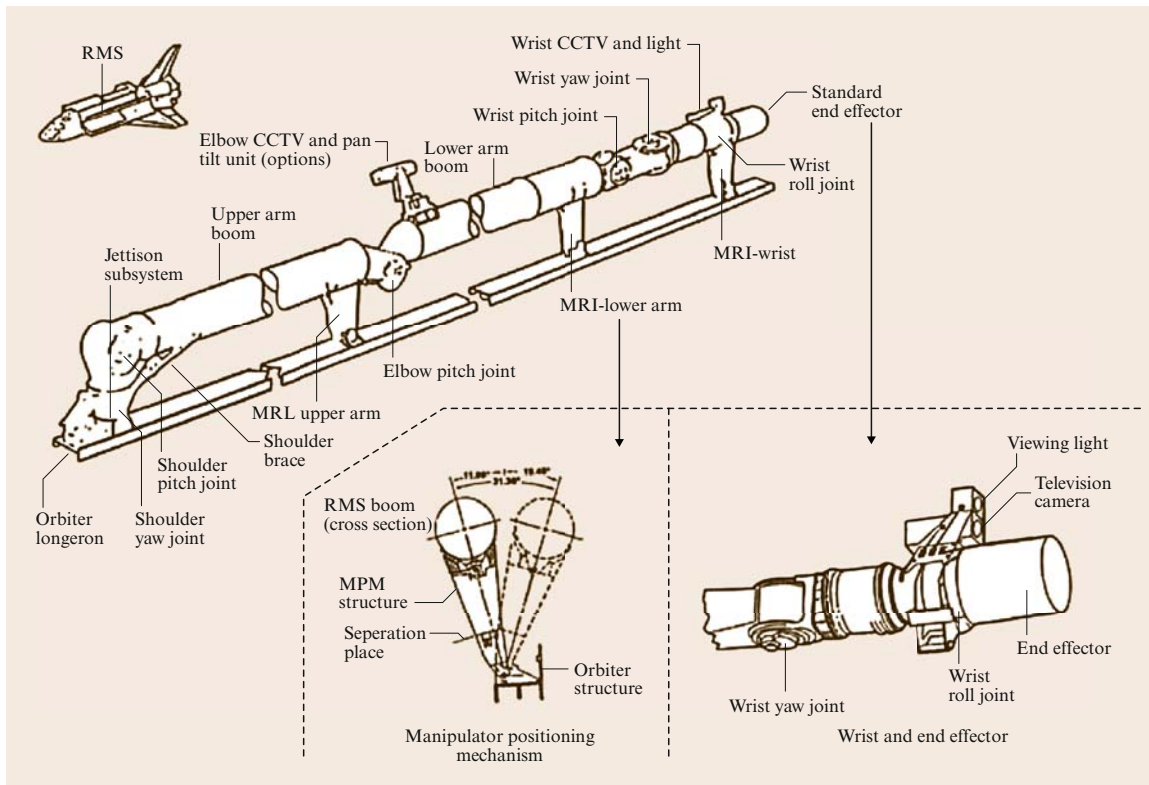


Fig. 45.2 The space shuttle remote manipulator system (SRMS) [45.2]; manipulator retention latch (MRL), manipulator positioning mechanism (MPM), remote manipulator system (RMS)

By attaching a foothold at the end point, the arm can serve as a mobile platform for an astronaut's extravehicular activities (Fig. 45.3).

After the Space Shuttle COLUMBIA accident during STS-107, the National Aeronautics and Space Administration (NASA) outfitted the SRMS with the orbiter boom sensor system (OBSS) – a boom containing instruments to inspect the exterior of the shuttle for damage to the thermal protection system [45.3]. The SRMS is expected to play a role in all future shuttle missions.

45.1.2 ISS-Mounted Manipulator Systems

The International Space Station (ISS) is the largest international technology project, with 15 countries making significant cooperative contributions. The ISS is an *out-post* of the human presence in space, as well as a *flying laboratory* with substantial facilities for science and engineering research. In order to facilitate various activities on the station, there are several robotic systems, some

of which are already operational, while others are ready for launch.

The *space station remote manipulator system* (SSRMS) or Canadarm 2, (Fig. 45.4), is the next generation of the SRMS, for use on the ISS [45.4]. Launched in 2001 during STS-100 (ISS assembly flight 6A), the SSRMS has played a key role in the construction and maintenance of the ISS both by assisting astronauts during EVAs and using the SRMS on the Space Shuttle to hand over a payload from the Space Shuttle to the SSRMS. The arm is 17.6 m long when fully extended and has seven degrees of freedom. Latching end-effectors, through which power, data, and video can be transmitted to and from the arm, are attached to both ends. The SSRMS is self-relocatable using an inch-worm-like movement with alternate grapples of *power data grapple fixtures* (PDGFs), which are installed over the station's exterior surfaces to provide the power, data, and video, as well as a foothold.

As another mobility aid for the SSRMS to cover wider areas of the ISS, a *mobile base system* (MBS) was

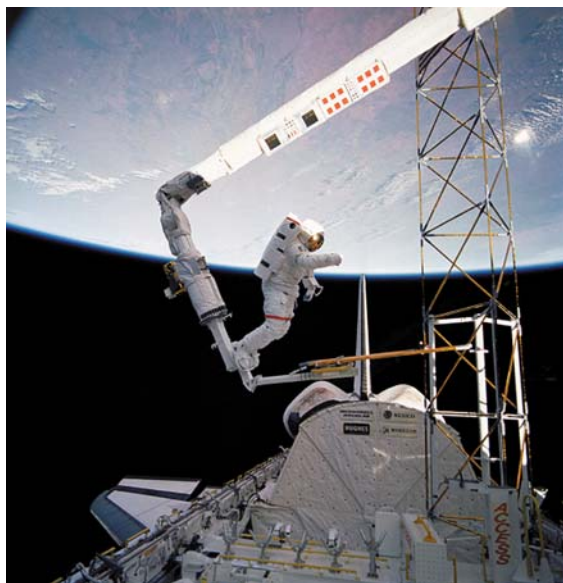


Fig. 45.3 Space shuttle remote manipulator system (SRMS) used as a platform for an astronaut's extravehicular activity in the shuttle cargo bay

added in 2002 by STS-111 (ISS assembly flight UF-2). The MBS provides lateral mobility as it traverses the rails on the main trusses [45.5].

The *special-purpose dexterous manipulator* (SPDM), or Dextre, which will be attached at the end of the SSRM, is a capable mini-arm system to facilitate the delicate assembly tasks currently handled by astronauts

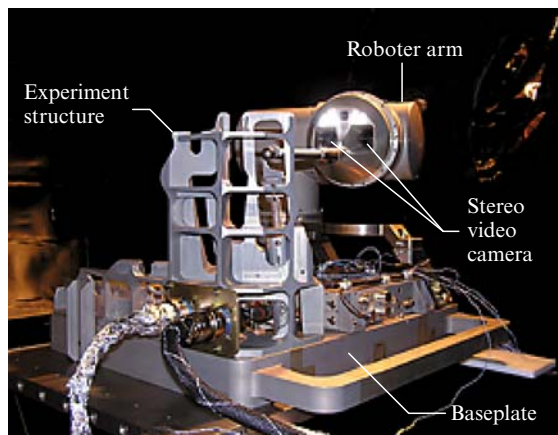


Fig. 45.5 Laboratory test bed for the ROKVISS manipulator system

during EVAs. The SPDM is a dual-arm manipulator system, where each manipulator has seven degrees of freedom and is mounted on a one-degree-of-freedom body joint. Each arm has a special tool mechanism dedicated to the handling of standardized *orbital replacement units* (ORUs). The arms will be teleoperated from a *robotic work station* (RWS) inside the space station [45.6].

The *European Space Agency* (ESA) will also provide a robotic manipulator system for the ISS, the *European robotic arm* (ERA), which be used mainly to work on the Russian segments of the station [45.7]. The arm is 11.3 m long and has seven degrees of freedom. Its basic configuration and functionality are similar to SSRMS [45.8].

As a precursor demonstration of the ERA, the *German Aerospace Agency* (DLR) has developed a 0.5 m-long, two-degree-of-freedom manipulator arm with a dedicated test bench, called *robotic components verification on the ISS* (ROKVISS), see Fig. 45.5. ROKVISS was launched by an unmanned Russian Progress transport vehicle in 2004 and installed on the outer platform of the Russian segment of the station in early 2005 [45.9]. Although the number of joints is small, challenging experiments of telepresence have been conducted in which human operators on the ground operate the arm based on stereovision images and bilateral, high-fidelity force feedback using a force-reflecting joystick on the ground, and a joint torque control system on board. The secondary goal of the experiment was the space qualification of the joint drives, which are the key components of DLR's torque-controlled lightweight robots [45.10].

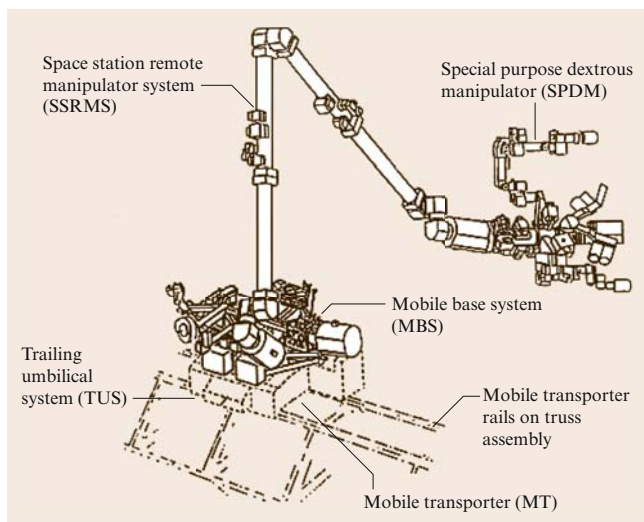


Fig. 45.4 Space station remote manipulator system (SSRMS) [45.4]

In Japan, the *Japanese experiment module remote manipulator system* (JEMRMS), shown in Fig. 45.6, has been developed by the *Japan space exploration agency* (JAXA) [45.11]. The arm will be attached to the Japanese module of the ISS. JEMRMS consists of two components: the main arm, a 9.9 m-long, six-degree-of-freedom arm and the small fine arm, a 1.9 m-long, six-degree-of-freedom arm. Unlike the SSRMS or the ERA, the main arm does not have self-relocation capability, but is fitted with a small fine arm, with which JEMRMS can form a serial 12-degree-of-freedom macro–micro manipulator system. After installation, the arm will be used to handle and relocate the components for the experiments and observations on the exposed facility [45.12, 13].

45.1.3 ROTEX

The *robot technology experiment* (ROTEX), developed by the German Aerospace Agency (DLR), is one of the important milestones of robotics technology in space [45.14]. A multisensory robotic arm was flown on the Space Shuttle COLUMBIA (STS-55) in 1993. Although the robot worked inside a work cell on the shuttle, several key technologies, such as a multisensory gripper, teleoperation from the ground and by the astronauts, shared autonomy, and time-delay compensation by the use of a predictive graphic display were successfully tested (Fig. 45.7).

45.1.4 ETS-VII

Engineering test satellite (ETS)-VII, shown in Fig. 45.8, was another milestone in the development of robotics technology in space, particularly in the area of satellite servicing. ETS-VII was an unmanned spacecraft developed and launched by the *National Space Development Agency of Japan* (NASDA, currently JAXA) in November 1997. A number of experiments were successfully conducted using a 2 m-long, six-degree-of-freedom manipulator arm mounted on its carrier (main) satellite.

The mission objective of ETS-VII was to test free-flying robotics technology and to demonstrate its utility in unmanned orbital operation and servicing tasks. The mission consisted of two subtasks: autonomous rendezvous/docking (RVD) and a number of robot experiments (RBT). The robot experiments included:

1. teleoperation from the ground with a large time delay
2. robotic servicing task demonstrations such as ORU exchange and deployment of a space structure

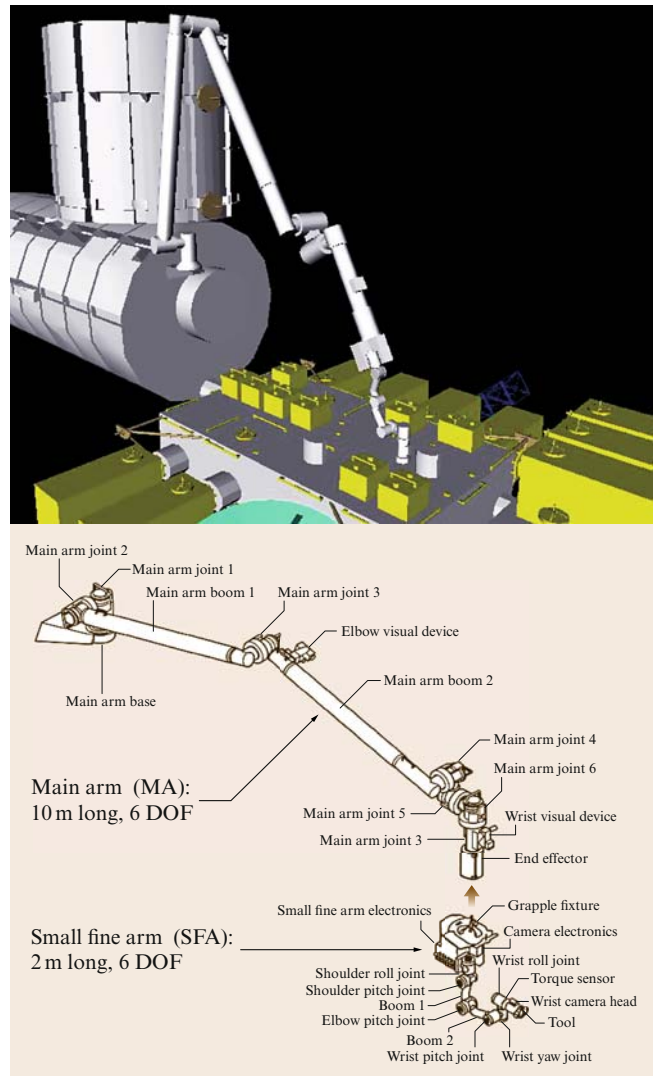


Fig. 45.6 Japan experiment module (JEM) on the ISS and the JEMRMS manipulator system

3. dynamically coordinated control between the manipulator reaction and the satellite attitude response
4. capture and berthing of a target satellite [45.15]

The communication time delay due to radio propagation (speed of light) is relatively small, for example, 0.25 s for a round trip to *geostationary Earth orbit* (GEO). However, to have a global coverage of communication in low-Earth-orbit operations, the signals are transmitted via multiple nodes including data relay satel-

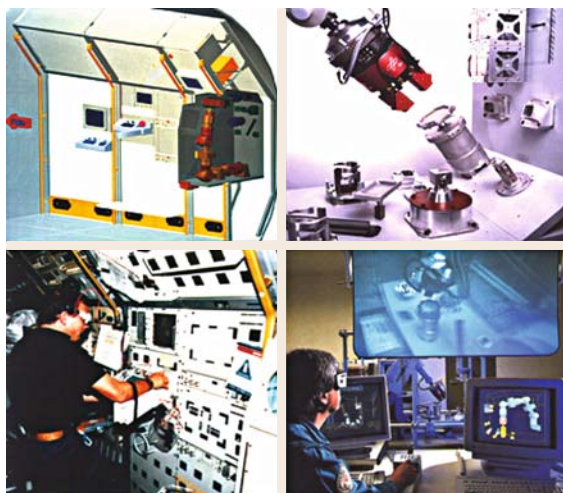


Fig. 45.7 ROTEX manipulator arm onboard the Spacelab D2 mission, the first remotely controlled space robot

lites located at GEO and ground stations. This makes the transmission distance longer, and even more additional delays are added at each node. As a result the cumulative delay becomes some seconds, actually 5–7 s in the case of the ETS-VII mission.

Opportunities for academic experiments were also opened to Japanese universities and European institutions (e.g., DLR and ESA) and important flight data were obtained that validate the concepts and theories for free-flying space robots [45.16, 17].

45.1.5 Ranger

Ranger is a teleoperated space robot being developed at the University of Maryland's Space Systems Lab-

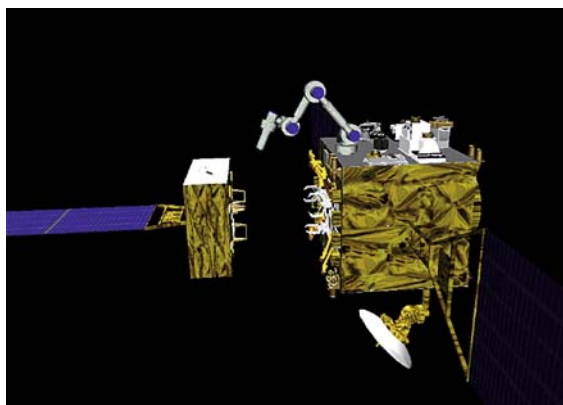


Fig. 45.8 Japanese engineering test satellite ETS-VII

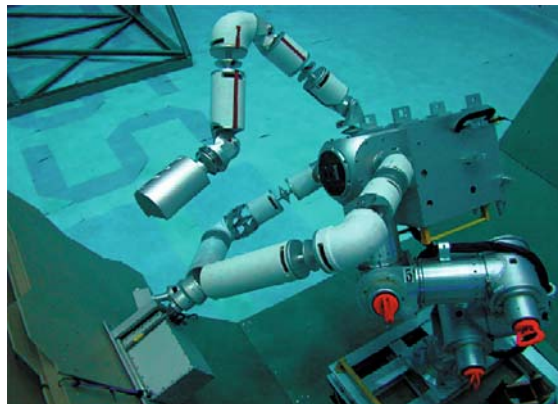


Fig. 45.9 Neutral buoyancy test of the Ranger telerobotic shuttle experiment

oratory [45.18]. Ranger consists of two seven-degree-of-freedom manipulators with interchangeable end-effectors to perform such tasks as changeout of *orbital replacement units (ORUs)* in orbit. Also discussed was the changeout of the *electronics controller unit (ECU)* of the Hubble space telescope, which previously required human EVA. A number of tests and demonstrations for servicing missions have been conducted at the University of Maryland neutral buoyancy facility (Fig. 45.9). Originally designed for a free-flying flight experiment, Ranger has been redesigned for a Space Shuttle experiment, but has not yet been manifested on a flight.

45.1.6 Orbital Express

The Orbital Express space operations architecture program is a DARPA program developed to validate the

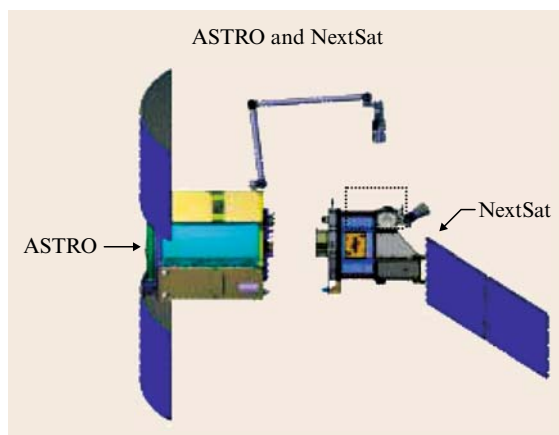


Fig. 45.10 Orbital Express flight mission configuration

technical feasibility of robotic on-orbit refueling and reconfiguration of satellites, as well as autonomous rendezvous, docking, and manipulator berthing [45.19]. The system consists of the *autonomous space transport robotic operations* (ASTRO) vehicle, developed by Boeing Integrated Defense Systems and a prototype modular next-generation serviceable satellite, NextSat, developed by Ball Aerospace. The ASTRO vehicle is equipped with a robotic arm to perform satellite capture and ORU exchange operations (Fig. 45.10).

After its launch in March 2007, various mission scenarios have been conducted. These scenarios include

1. visual inspection, fuel transfer, and ORU exchange on NextSat using ASTRO's manipulator arm when both spacecrafts are connected
2. separation of NextSat from ASTRO, orbital maneuvers by ASTRO and fly-around, rendezvous, and docking with NextSat
3. capture of NextSat using ASTRO's manipulator arm

These scenarios were successfully completed by July 2007, with ASTRO's onboard autonomy using onboard cameras and advanced video guidance system.

45.2 Historical Developments and Advances of Surface Robotic Systems

The research on surface exploration rovers began in the mid-1960s, with an initiative (that never flew) for an unmanned rover for the Surveyor lunar landers and a manned rover (Moon buggy) for the human landers in the United States. In the same period, research and development began for a teleoperated rover named Lunokhod in the Soviet Union. Both the Apollo manned rover and the Lunokhod unmanned rover were successfully demonstrated in the early 1970s on the Moon [45.20]. In the 1990s the exploration target had expanded to Mars and in 1997, the Mars Pathfinder mission successfully deployed a microrover named Sojourner that safely traversed the rocky field adjacent to the landing site by autonomously avoiding obstacles [45.21, 22]. Following on from this success, today autonomous robotic vehicles are considered indispensable technology for planetary exploration. The twin Mars exploration rovers, Spirit and Opportunity, were launched in 2003 and have had remarkable success in terms of remaining operational in the harsh environment of Mars for over four years. Each has traveled more than 5000 m and has made significant scientific discoveries using onboard instruments [45.23, 24].

45.2.1 Teleoperated Rovers

The first remotely operated robotic space surface vehicle was Lunokhod (Fig. 45.11) [45.20]. Lunokhod 1 landed on the Moon on 17 November 1970 as a payload on the lander Luna-17 and Lunokhod 2 landed on the Moon on 16 January 1973. Both were eight-wheeled skid-steered vehicles having a mass of about 840 kg, where almost all components were in a pressurized *bathtub* thermal enclosure with a lid that closed over the tub to allow it to survive the deep cold (≈ 100 K) of the long lunar

nights using only the heat emitted by small pellets of radioisotope. On the inside of the lid were solar arrays which recharged batteries during the day as required to maintain the operation of the vehicle. Lunokhod 1 operated for 322 Earth days, traversing over 10.5 km during that period and returned over 20 000 TV images, 200 high-resolution panoramas, and the results of more than



Fig. 45.11 Lunokhod

500 soil penetrometer tests and 25 soil analyses using its X-ray fluorescence spectrometer. Lunokhod 2 operated for about 4 months, having traversed more than 37 km, with the mission officially terminating on 4 June 1973. It has been reported that Lunokhod 2 was lost prematurely when it began sliding down a crater slope and hasty commands were sent in response which ultimately caused the end of the mission.

Each of the eight wheels on the Lunokhod vehicles were 0.51 m in diameter and 0.2 m wide, giving an effective ground pressure of less than 5 kPa based on an assumed sinkage of 3 cm. Each wheel had a brush-type direct-current (DC) motor, a planetary gear reduction, a brake and a disengagement mechanism allowing it to free-wheel in the event of some problem with the motor or gears. The mobility commands to the vehicle included two speeds forward or backward, braking, and turning to the right or the left either while moving or in place.

The vehicles had both gyroscope- and accelerometer-based tilt sensors which could automatically stop the vehicle in the event of excessive tilt of the chassis. Typical mobility commands specified a time duration over which the motors would run and then stop. Precision turning commands specified the angle through which the vehicle should turn. These commands were terminated when the specified turn angle had been reached according to the heading gyroscope. Odometry was determined by a ninth *small wheel* which was unpowered and lightly loaded and used only to determine over-the-ground distance. There was an onboard current-overload system and motor currents, pitch and roll measurements, and distance traversed and many component temperatures were telemetered to the ground operators.

The Lunakod crew consisted of a driver, a navigator, a lead engineer, an operator for steering the *pencil-beam* communication antenna, and a crew commander. The driver viewed a monoscopic television image from the vehicle and gave the appropriate commands (*turn*, *proceed*, *stop* or *back up*) along with their associated parametric value in terms of duration or angle. The navigator viewed displays of telemetry from the vehicle's *course gyroscope*, *gyrovertical sensor*, and odometer and was responsible for calculating the trajectory of the vehicle and laying out the route to be followed. Thus the driver was responsible for vehicle stability about its center of mass and the navigator was responsible for the trajectory of that center of mass. The lead engineer (assisted by many specialists as required), was responsible for assessing the health of the onboard systems. The lead engineer provided both routine updates on energy supply, thermal conditions, etc. as well as possible

emergency alerts such as extreme motor currents or chassis tilt. The pencil-beam antenna operator oversaw the functioning of an independent ground-based closed-loop control system that servoed the antenna to always point at Earth, independent of the vehicle motion. The crew commander supervised the implementation and execution of the overall plan, gave any detailed commands for making actual contact with the surface (e.g., by the penetrometer) and also could override any command to the vehicle as he viewed the same information as the driver.

The entire driving system was tested extensively prior to the Lunokhod 1 mission at a *lunodrome* with simulated lunar terrain which proved to be more challenging than that actually encountered during the Lunokhod 1 mission. Despite this, the operators of Lunokhod 1 said they *encountered a dangerous situation* (unforeseen entrance into a crater, rolling onto a rock, etc.) slightly more often than once per kilometer. This was attributed to inadequate driving experience, the modest quality of the television images and the poor illumination conditions on the Moon. The driving direction was often selected primarily to give the best images; even so, the operators reported *fictional dangers* caused by varying illumination conditions. In the first three months (lunar days) of operation, the vehicle traversed 5224 m in 49 h of driving using 1695 driving commands, including about 500 turns. Sixteen signals were sent for protection against excessive tilt during that time; approximately 140 craters were traversed at maximum slope angles of 30°.

With the approximately 2.6 s speed-of-light delay, the operators stated that “control experience confirmed the desirability of movement in a starting–stopping regime with mandatory stopping each few meters.” The soil properties were found to “differ substantially even in terrain sectors not very distant from one another.” The soil penetrometer determined that the upper layer of regolith varied from a stiffest where the penetrometer required about 16 kg (Earth weight) of force to penetrate about 26 mm, to a weakest measurement where only 3 kg of Earth weight caused a penetration of about 39 mm. The cone penetrometer had a base diameter of 50 mm and a cone height of 44 mm. Thus the upper layer of regolith had a rate of increase of load-bearing strength ranging from about 400 kPa/m for the weakest soil to about 3 MPa/m for the stiffest soil. Crater walls and the immediate ejecta blanket around craters generally exhibited the weakest soil. Below 5–10 cm of penetration depth, the regolith generally became rapidly stiffer. The mean value of wheel slippage for the first three lunar days was about 10%. On horizontal terrain,

the slippage was 0–15%, depending on surface irregularities and ground inhomogeneity. On crater slopes the slip increased to 20–30%. The specific resistance of the Lunokhod wheels was generally in the range 0.05–0.25, while the specific free traction (the ratio of traction to weight) was in the range 0.2–0.41. The crater distribution in the area explored by Lunokhod 1 was found to be closely approximated by the formula $N(D) = AD^{-\delta}$, where $N(D)$ is the number of craters larger than diameter D meters per hectare of lunar surface, A is a scale factor found to be about 250 and δ is the distribution exponent, found to be about 1.4 [45.20].

45.2.2 Autonomous Rovers

In the mid-1960s, research began on a lunar rover at the [US Jet Propulsion Laboratory \(JPL\)](#) in Pasadena, California, when it was proposed to put a small rover on the Surveyor lunar landers. These landers were led by [JPL](#) (based on a system-level contract to Hughes) and were designed to land softly on the Moon to establish the safety of such a landing prior to the Apollo landers with humans aboard. At the time it was speculated (notably by T. Gold) that the Moon might be covered in a thick layer of soft dust that would *swallow* any lander. In 1963 [JPL](#) issued a contract to build a small rover concept prototype in support of the Surveyor program to the General Motors Defense Research Laboratories in Goleta, California. That GM facility had recently hired M.G. (Greg) Bekker, who was considered *the father of off-road locomotion*, having written several seminal textbooks on the subject and having introduced many of the key concepts relating soil properties to off-road vehicle performance that are still used today [45.25, 26] (Sect. 45.3.12).

Bekker and his team proposed an articulated six-wheeled vehicle based on a novel three-cab configuration with an axial spring-steel suspension. This vehicle exhibited remarkable mobility, being able to climb vertical steps up to three wheel radii high and crossing crevasses three wheel radii wide. Notable people working with Bekker were Farenc Pavlics, who went on to lead the development of the mobility system for the Apollo lunar rover (under contract from Boeing), and Fred Jindra, who developed the underlying equations describing the mobility of the six-wheeled articulated vehicle that were later used by Don Bickler in conceiving the *rocker bogie* chassis used on the Sojourner and the Mars exploration rovers. Bekker and his team proposed the six-wheeled articulated vehicle after experimenting with many types of vehicles, including multitracked ve-

hicles, screw-type vehicles (for fine-powdered terrain) and others. In scale-model testing the six-wheeled vehicle demonstrated superior performance over both soft and rocky terrain.

They built and delivered two vehicles that were about 2 m long with approximately 0.5 m wheel diameters. Those vehicles were used in testing throughout the 1960s and early 1970s to conduct simulated operations to determine how such vehicles could actually be used on the Moon. One key issue was that the speed-of-light round trip from the Moon (about 3 s) precluded direct *driving* of the vehicle. Perhaps most annoying was the fact that, during vehicle motion, the highly directional radio antenna used to communicate with Earth would lose its pointing and so communications would briefly be lost. This meant that operators driving the rover would be confronted with a series of still images, instead of a stream of moving images. It was quickly realized that much of an operator's *situation awareness* and depth perception needed to drive a vehicle with a monocular camera comes from motion. It was very difficult to drive from frozen monocular images. A crude form of stereo was incorporated where the camera mast was raised and lowered slightly and the operator could switch back and forth between the two views.

Following the successful landing of several of the Surveyor spacecraft and the discovery that all landing sites seemed to have relatively firm soil, it was concluded that the Surveyor lunar rover was not needed. As a result, the prototype was used for research into the early 1970s and subsequently restored for use again in research in the 1980s, becoming the first vehicle to be outfitted with *waypoint navigation* of the sort later used on the Sojourner and [MER](#) missions.

About the time that the stationary 1976 Viking landers were conceived and developed, [JPL](#) began the 1984 Mars Rover effort. (1984 was an energetically favorable launch opportunity from Earth to Mars and the next likely major mission opportunity after Viking.) Two test bed vehicles were developed, a *software* prototype and a *hardware* prototype. The software prototype had a *Stanford* arm, designed by Vic Scheinman (who went on to design the Unimation PUMA arm and many other famous early robotic devices). This was the only 1.5-scale Stanford arm ever built. Antal (Tony) *Beczcy* became well known in robotics for solving the kinematics of this arm, one of if not the first full kinematics ever done in robotics up to that time [45.27]. A stereo pan-tilt head was implemented, equipped with the first solid-state cameras to become available. A number of very important works were published in

the 1977 International Joint Conference on Artificial Intelligence [45.28]. The first *hand-eye locomotion* coordination was done with this vehicle, where a rock was designated in a stereo image and the vehicle maneuvered autonomously to a point where the arm could reach out and pick it up. One of the first demonstrations of *pin in hole* insertion and other *dexterous* manipulations were also done with this system in the 1970s.

The *hardware* prototype was built using elastic *loop wheels* made by *Lockheed* [45.29]. The vehicle was battery powered and controlled via a handheld remote-control (RC) unit of the type used by hobbyists.

In late 1982 the JPL had a contract with the US Army to study the use of robotic support vehicles. During this study, Brian Wilcox at JPL proposed a technique to reduce the need for a real-time video link or high-bandwidth communication channel between the vehicle and the operator. This technique (which became known as *computer-aided remote driving* or *CARD*) [45.30], re-

quired the transmission of a single stereo image from the vehicle to the operator, so the operator could designate waypoints in that image using a three-dimensional (3-D) cursor. By the use of a single stereo image instead of a continuous stream of monocular images, the amount of information that needed to be transmitted by the vehicle was reduced by orders of magnitude. JPL first demonstrated *CARD* on the resurrected *surveyor lunar rover vehicle* (SLRV, which had been painted baby-blue and so became known as the *Blue Rover*, see Fig. 45.12a), and later on a modified Humvee. During field tests in the Mojave desert Dn 1988, *CARD* was demonstrated on the Humvee with path designations of 100 m per stereo image and with time to designate each path of only a few seconds.

As the *CARD* work was ongoing, an internally funded effort at JPL demonstrated a concept called *semi-autonomous navigation* (SAN). This concept involved humans on Earth designating *global* paths using maps of the sort that could be developed from orbiter imagery and then having the vehicle autonomously refine and execute a *local* path that avoids hazards. The moderate success of that effort led to a NASA-funded effort, leading to the development of a new vehicle, called Robby (Fig. 45.12b). Robby was a larger vehicle that could support the onboard computing and power needed for untethered operation. (The SLRV had been tethered to a VAX 11/750 minicomputer over a 500 m tether during arroyo field testing of *CARD*.) For the first time (in 1990), an autonomous vehicle had made a traverse through an obstacle field that was faster than a rover could have done on Mars using human path designation done on Earth.

However, Robby had a severe *public relations* problem – it was perceived as too large. Of course none of the computers or power systems had been miniaturized or lightweighted – it was composed entirely of the lowest-cost components that could do the job. However, because it was the size of a large automobile, observers and NASA management got the impression that future rovers would be car-sized or even truck-sized vehicles. This was compounded by the *Mars rover sample return* (MRSR) study done by JPL in the late-1980s, which suggested a mass for the rover of 882 kg. An independent study of the MRSR study by Science Applications International, Inc. (SAIC) estimated the overall cost at \$13B. When word of this *outrageous* price tag filtered around NASA Headquarters and into the Congressional Staff, MRSR was summarily killed. Robby died along with it. At about the same time, NASA funded Carnegie Mellon University to develop *Ambler*

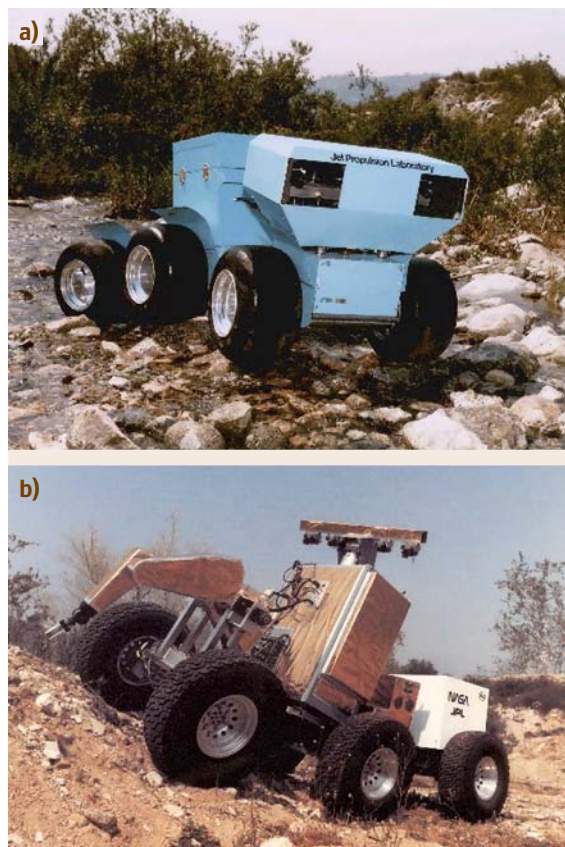


Fig. 45.12a,b Six-wheel articulated body rovers developed by JPL (a) SLRV and (b) Robby

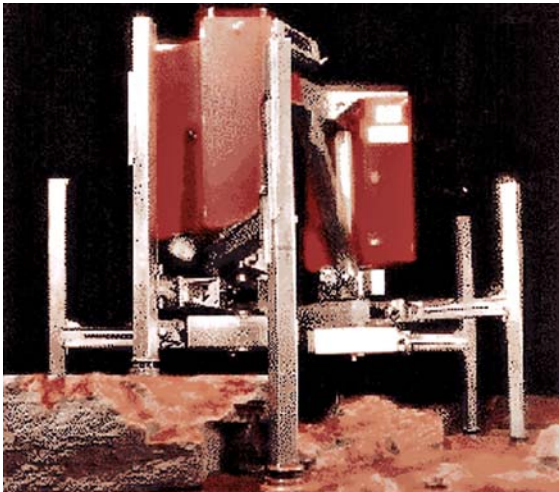


Fig. 45.13 Ambler

(Fig. 45.13), a large walking robot that was able to autonomously choose safe footfall locations, also as a test bed Mars Rover [45.31,32]. Ambler had a similar public-relations problem, being about the same mass as Robby, that the NASA management community was very skeptical that such large systems could affordably be flown to Mars. Both Robby and Ambler had all-onboard power and computing systems, which at that time were not sufficiently miniaturized to make autonomous rovers credible for actual flight missions. *Moore's law* was not only causing the computing technology to become miniaturized at a high rate, but also the energy required per computing instruction was dropping rapidly. This meant that early systems devoted most of their power to computing rather than to motive power. Later systems,

such as the Mars exploration rovers, have a more nearly equal balance between power for mobility and power for computation. Future systems will presumably devote the majority of their power to mobility as opposed to computation.

Soon thereafter, the *Mars environmental survey* (MESUR) mission set was proposed as a lower-cost alternative to a sample return mission. The MESUR Pathfinder mission was proposed as a first test of what was envisioned as a network of 16–20 surface stations to provide global coverage of Mars. A small rover was proposed to the Mars science working group [45.33,34]. A very short-term development effort culminated in a demonstration in July 1992 of a ≈ 4 kg rover that could move to directed points on the surface nearby a lander using stereo designation of waypoints in a 3-D display of frozen images taken from a lander mast camera pair (Fig. 45.14). This demonstration was sufficiently successful that a similar rover was manifested for the Mars Pathfinder mission. The Pathfinder rover (Fig. 45.15) was later named Sojourner, and became the first autonomous vehicle to traverse the surface of another planet, using a hazard detection and avoidance system to move safely between waypoints through a rock field [45.21, 22]. The hazard detection system avoided obstacles and was also used to position the vehicle accurately in front of rocks. Sojourner operated successfully for 83 Martian days (until the failure of the lander, which was acting as a communications relay between the rover and Earth). Sojourner examined approximately a dozen rock and soil samples with its alpha–proton–X-ray spectrometer, which gives the elemental composition of the rocks and soil. The success of Sojourner led directly to the decision to build the twin Mars exploration rovers



Fig. 45.14 Rocky 4



Fig. 45.15 The Pathfinder rover, Sojourner

launched in 2003. Both Sojourner and the MER rovers use waypoint designation in stereo images by the human operator together with autonomous hazard detection and avoidance to keep the rovers safe if they should wander off the designated path.

During the 1992–1993 summer season in Antarctica, the Dante I robot, built by Carnegie-Mellon University and funded by NASA, attempted to rappel into the caldera of the active volcano Mt. Erebus. Dante was a walking robot and was the first serious attempt to make a robot rappel down a grade that was too steep to traverse using purely frictional contact. Unfortunately, the extreme cold (even in the summer) compounded by human error caused a kink in the fiber-optic umbilical to snag going through an eyelet, breaking the high-bandwidth fiber communications on which the system depended. The fiber could not be repaired in the field and so the mission was aborted. Undaunted, in the summer of 1994, Dante II (Fig. 45.16) made a successful rappel into the caldera of Mt. Spurr in Alaska, exploring the active vents on the crater floor in a way that would be unsafe in the extreme if done by human explorers. The Dante robot series demonstrated that rappelling, especially when combined with legged locomotion, allows

robots to conduct exploration to extremely hazardous sites in ways that humans cannot.

In 1984 NASA started the Telerobotics Research program [45.35,36]. This program demonstrated various aspects of on-orbit assembly, maintenance, and servicing. Some highlights of this activity were the automated tracking and grapping of a free-spinning satellite (suspended with a counterweight and gimbal for realistic reactions under external forces), connection of a flight-like fluid coupler and many *busy box* functions such as door opening, threaded fastener mating and demating, use of power tools, dual-arm manipulation of a simulated hatch cover and flexible thermal blanket, etc. by various control approaches ranging from force-reflecting teleoperation to fully autonomous sequences. This activity ended in about 1990.

45.2.3 Research Systems

There have been many mobile robots built by government, university, and industrial groups whose objective was to develop new technologies for planetary surface exploration or to excite students or young engineers about the possibilities in that area. Carnegie Mellon University developed the Ambler, Dante, Nomad, Hyperion, Zoe, and Icebreaker robot series. The *Jet Propulsion Laboratory (JPL)*, Draper Labs, MIT, Sandia National Lab, and Martin-Marietta (later Lockheed-Martin) each built more than one planetary surface robot test bed. The Marsokhod chassis built by VNII Transmach of St. Petersburg, Russia was used by research groups there and also in Toulouse (LAAS and CNRS) [45.37, 38] as well as the NASA Ames Research Center and McDonnell Douglas Corporation (later part of the Boeing Company) in the US.

These research platforms have been used for two basic avenues of research. One is to perfect safe driving techniques on planetary surfaces, despite the speed-of-light latencies inherent in robotic exploration of the planets. This includes the waypoint navigation technology developed at JPL in the 1980s, where frozen stereo images are used to plan a possibly lengthy series of waypoints or activity sites and then executed with various sorts of reflexive hazard avoidance or safing techniques, such as used on the Sojourner rover on Mars in 1997. The other is to develop higher-level autonomy for improved science data return or mission robustness. Technologies in this latter category include mission planners that attempt to optimize routes and activity sequences based on time, limits to peak power, total energy, expected temperature, illumination angles, availability of

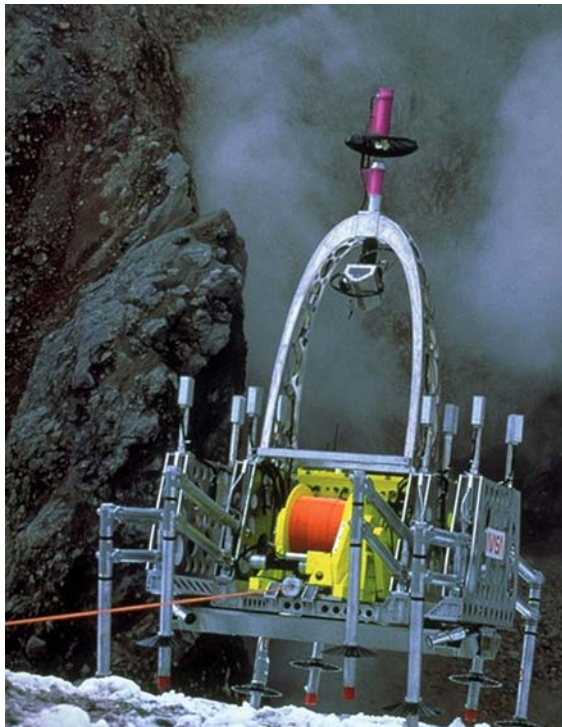


Fig. 45.16 Dante II at Mt. Spurr in Alaska

communications and others. Automated classification of possible science targets based on clustering of spectral data, figure-ground segmentation of rocks and other approaches have been attempted with some success. At the time of this writing some of these technologies have been uploaded to the twin Mars exploration rovers *Spirit* and *Opportunity* (Fig. 45.18) [45.23], including automated detection of temporary events of scientific interest such as dust-devils and clouds [45.39].

45.2.4 Sensing and Perception

In the 1980s, most planetary rover sensing research was based on laser ranging or stereo vision. Stereo vision was too computationally intensive for early low-power, radiation-hard processors, so the Sojourner Mars rover used a simple form of laser ranging to determine which areas were safe to traverse. Between the launch of Sojourner (1996) and the Mars exploration rovers (2003) sufficient progress had been made in radiation-hardened flight processors that stereo vision was used for hazard detection on Mars exploration rovers (*MER*) [45.40], mostly in experiments conducted with the Rocky-7 rover (Fig. 45.17). This allowed much larger numbers of range points to be incorporated into the hazard-detection algorithm (thousands of points, instead of the 20 discrete range points used by Sojourner). Perception of hazards on Sojourner was based on simple computations of average slope and roughness over the 4×5 array of range measurements, as well as the maximum height differences.

Each *MER* rover used a more sophisticated evaluation of the safety of the rovers along a large number of candidate arcs from its current location. Many other algorithms for the perception of terrain hazards have been used with reasonable success by various organizations. Today it is probably fair to say that the *unsolved problems* lie not in the area of geometric hazards (e.g., hazards that can be evaluated completely based on accurate knowledge of the shape of the terrain) but rather in the area of nongeometric hazards (e.g., hazards where uncertainties in the load-bearing or frictional properties of the terrain determine the safety of a proposed traverse). Accurate estimation of the load-bearing or friction properties of terrain by remote sensing is a very challenging task that will not be completely *solved* anytime soon, if ever.

45.2.5 Estimation

Most estimation for planetary surface exploration relates to the internal state of the robot or its position, pose, and



Fig. 45.17 Rocky 7

kinematic configuration with respect to the environment. Internal state sensors such as encoders on any active or passive articulations in the vehicle are used, along with a kinematic model and inertial sensors such as accelerometers and gyroscopes, to estimate the pose of the vehicle in inertial space. Perceptual algorithms such as surface reconstruction from clouds of range points as developed by stereo vision put terrain geometry estimates into this same representation. Heading in inertial space is generally the most difficult to estimate reliably, due to the lack of navigation aids such as the global positioning system or any easily measured heading reference such as a global magnetic field. Integration of rate-gyro data is used to maintain local attitude during motion while accu-



Fig. 45.18 The Mars exploration rovers, spirit and opportunity, with a manipulator arm in front

rate estimation of the rotation axis for Mars is possible by long integrations of three-axis rate-gyro data while the vehicle is stopped. Similar approaches are probably not feasible for the Moon, because of its slow rotation rate. Imaging of the solar disk or constellations of stars at precisely known times can be combined with stored models of the rotation of the planet to allow accurate estimation of the complete pose of the vehicle in inertial space. Kalman filtering or related techniques are generally employed to reduce the effects of measurement noise.

45.2.6 Manipulators for In Situ Science

The Mars exploration rovers were the first planetary exploration vehicles to have general-purpose manipulators. (Lunokhod had a single-purpose soil penetrometer and Sojourner had a single-degree-of-freedom device to place an alpha-proton-X-ray spectrometer in direct contact with the terrain.) The **MER** arms each have five degrees-of-freedom and a reach of over 1 m. Accurate gravity-sag models of the lightweight arm allow the precise position to be predicted in advance of any command to deploy an instrument and contact sensors allow the

arm to stop before any excessive forces build up in the relatively flexible arm. Future arms for planetary surface operations, especially any proposed assembly, maintenance or servicing tasks as part of the proposed lunar outpost, will require force sensing to protect the stronger but much more rigid arms from damage, as well as to allow controlled forces to be applied to workpieces. Of course there is a huge body of knowledge associated with industrial robot arms and undersea robotics (e.g., for the offshore oil industry), but such arms are generally very heavy, fast, and stiff compared with credible systems for planetary surface use. Delicate force control has rarely been applied to industrial settings. Space hardware is necessarily very lightweight and so both the arms and the workpieces will need to have well-resolved force sensing and control to prevent damage to one or both. Because of severe limits on both mass and power, space manipulation tends to be slow. Historically, this means the gear ratio between each motor and the corresponding output shaft is very large, making the use of motor current as an estimator for output torque very problematic. Other low-mass and robust means for accurate sensing of applied forces in the space environment are needed.

45.3 Mathematical Modeling

45.3.1 Space Robot as an Articulated-Body System

Broadly speaking, both on-orbit manipulators and surface mobile robots are considered to be common articulated body systems with a moving base. One point that clearly distinguishes them from other ground-based robots, such as industrial manipulators, is the existence of a moving base. The robotic systems discussed in this chapter consist of one or multiple articulated limbs mounted on a base body that has a dynamic coupling with these limbs. Typical styles of such moving base systems are categorized into the following groups [45.41].

Free-Floating Manipulator Systems

A space free-flyer that has one or more manipulator arms, as illustrated schematically in Fig. 45.19a, is a typical example of this group. When operating the manipulator arm(s), the position and orientation of the base spacecraft fluctuates due to the manipulator reaction. The kinetic momentum of the system is conserved if no external forces or moments are applied and the conservation law for this system governs the reaction

dynamics. Coordination or isolation between the base and manipulator dynamics is key to advanced motion control.

Macro-Micro Manipulator Systems

A robotic system that comprises a relatively small arm (micro arm) for fine manipulation mounted on a relatively large arm (macro arm) for coarse positioning, is called a macro-micro manipulator system. The **SSRMS** (Canadarm2) and the **SPDM** (Dextre) system, as well as the **JEMRMS** on the Japanese module of the **ISS**, are good examples. Here, the connecting interface at the end point of the macro arm or the root of the micro arm is modeled as the base body (Fig. 45.19b). A free-flying space robot may be treated in this group when its base body is actively controlled by actuators that produce external forces and moments, such as gas-jet thrusters. In this case, these actuators can be modeled as a virtual macro arm [45.42].

Flexible-Based Manipulator Systems

If the macro arm behaves as a passive flexible (elastic) structure in a macro-micro manipulator system, the

system is considered to be a flexible-based manipulator (Fig. 45.19c). Such a situation can be observed in the operation of the ISS, when the SSRMS is servo or brake locked after its coarse operation. Here, the issue is that the base body (the root of the micro arm or the end of the macro arm, according to the definition above) is subject to vibrations that will be excited by the reaction of the micro arm.

Mobile Robots with Articulated Limbs

Mobile robots for surface locomotion have the same structure in terms of the dynamics equation as the above groups. This group includes wheeled vehicles, walking (articulated limb) robots, and their hybrids. In a wheeled vehicle, suspension mechanisms, if any, are also modeled as articulated limb systems. The forces and moments yielded by contact with ground or planetary surface govern the motion of the system (Fig. 45.19d)

45.3.2 Equations for Free-Floating Manipulator Systems

Let us consider first a free-floating system with a single or multiple manipulator arm(s) mounted on a base spacecraft. The base body, termed link 0, is floating in inertial space without any external forces or moments. At the end point of the arm(s), external forces/moments may apply. For such a system, the equation of motion is expressed as follows

$$\begin{pmatrix} \mathbf{H}_b & \mathbf{H}_{bm} \\ \mathbf{H}_{bm}^\top & \mathbf{H}_m \end{pmatrix} \begin{pmatrix} \ddot{\mathbf{x}}_b \\ \ddot{\boldsymbol{\phi}} \end{pmatrix} + \begin{pmatrix} \mathbf{c}_b \\ \mathbf{c}_m \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \boldsymbol{\tau} \end{pmatrix} + \begin{pmatrix} \mathbf{J}_b^\top \\ \mathbf{J}_m^\top \end{pmatrix} \mathbf{F}_h. \quad (45.1)$$

The kinematic relationship among \mathbf{x}_h , \mathbf{x}_b , and $\boldsymbol{\phi}$ is expressed using Jacobian matrices as

$$\dot{\mathbf{x}}_h = \mathbf{J}_m \dot{\boldsymbol{\phi}} + \mathbf{J}_b \dot{\mathbf{x}}_b, \quad (45.2)$$

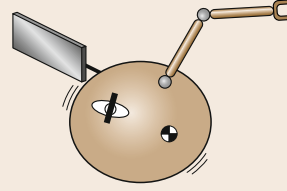
$$\ddot{\mathbf{x}}_h = \mathbf{J}_m \ddot{\boldsymbol{\phi}} + \dot{\mathbf{J}}_m \dot{\boldsymbol{\phi}} + \mathbf{J}_b \ddot{\mathbf{x}}_b + \dot{\mathbf{J}}_b \dot{\mathbf{x}}_b, \quad (45.3)$$

where the symbols are:

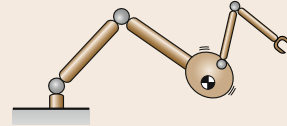
- $\mathbf{x}_b \in \mathbb{R}^6$: position/orientation of the base
- $\boldsymbol{\phi} \in \mathbb{R}^n$: joint angle of the arm
- $\mathbf{x}_h \in \mathbb{R}^{6k}$: position/orientation of the end point(s)
- $\boldsymbol{\tau} \in \mathbb{R}^n$: joint torque of the arm
- $\mathbf{F}_h \in \mathbb{R}^{6k}$: external forces/moments on the end point(s)
- n : number of total joints
- k : number of manipulator arms

and \mathbf{H}_b , \mathbf{H}_m , and \mathbf{H}_{bm} are inertia matrices for the base body, manipulator arm, and the coupling between

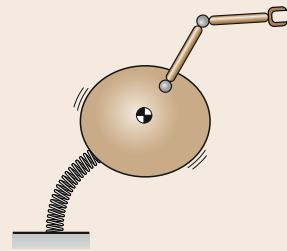
a) Free-floating manipulator system



b) Macro-micro manipulator system



c) Flexible structure based manipulator system



d) Surface locomotive robot system

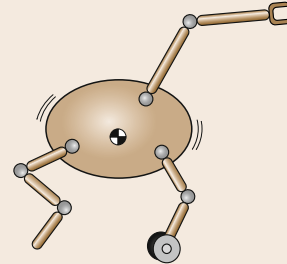


Fig. 45.19a–d Four basic types for moving base robots: (a) Free-floating manipulator system, (b) macro–micro manipulator system, (c) flexible-structure-based manipulator system, (d) surface locomotive robot system

the base and the arm, respectively, and \mathbf{c}_p and \mathbf{c}_q are nonlinear Coriolis and centrifugal forces, respectively.

For a free-floating manipulator in orbit, the gravity forces exerted on the system can be neglected and so the nonlinear term becomes $\mathbf{c}_b = \dot{\mathbf{H}}_b \dot{\mathbf{x}}_b + \dot{\mathbf{H}}_{bm} \dot{\boldsymbol{\phi}}$. Integrating the upper set of equation in (45.1) with respect to time, we obtain the total momentum of the system as

$$\mathcal{L} = \int \mathbf{J}_b^\top \mathbf{F}_h dt = \mathbf{H}_b \dot{\mathbf{x}}_b + \mathbf{H}_{bm} \dot{\boldsymbol{\phi}}. \quad (45.4)$$

For the case in which reaction wheels are mounted on the base body, they are included as additional manipulator arms.

45.3.3 Generalized Jacobian and Inertia Matrices

From (45.2) and (45.4), the coordinates of the manipulator base $\dot{\mathbf{x}}_b$, which are passive and unactuated coordinates, can be eliminated, as follows

$$\dot{\mathbf{x}}_h = \hat{\mathbf{J}}\dot{\boldsymbol{\phi}} + \dot{\mathbf{x}}_{h0}, \quad (45.5)$$

where

$$\hat{\mathbf{J}} = \mathbf{J}_m - \mathbf{J}_b \mathbf{H}_b^{-1} \mathbf{H}_{bm} \quad (45.6)$$

and

$$\dot{\mathbf{x}}_{h0} = \mathbf{J}_b \mathbf{H}_b^{-1} \mathcal{L}. \quad (45.7)$$

Since \mathbf{H}_b is the inertia tensor of a single rigid body (the manipulator base), it is always positive definite and so its inverse exists.

The matrix $\hat{\mathbf{J}}$ was first introduced in [45.43,44] and is referred to as the *generalized Jacobian matrix*. In its original definition, it was assumed that no external forces/moments acted on the system. If $\mathbf{F}_h = \mathbf{0}$, then the term $\dot{\mathbf{x}}_{h0}$ becomes constant and, in particular, if the system has zero initial momentum $\dot{\mathbf{x}}_{h0} = \mathbf{0}$, then (45.5) becomes simple. However, note that in the derivation of (45.6), zero or constant momentum is not a necessary condition.

Using this matrix, the manipulator hand can be operated under a resolved motion-rate control or resolved acceleration control in inertial space. Thanks to the generalized Jacobian, although the reactive base motion occurs during the operation, the hand is not disturbed by the reactive motion.

From the upper and lower sets of equations in (45.1), $\ddot{\mathbf{x}}_b$ can be eliminated to obtain the following expression

$$\hat{\mathbf{H}}\ddot{\boldsymbol{\phi}} + \hat{\mathbf{c}} = \boldsymbol{\tau} + \hat{\mathbf{J}}\mathbf{F}_h, \quad (45.8)$$

where

$$\hat{\mathbf{H}} = \mathbf{H}_m - \mathbf{H}_{bm}^T \mathbf{H}_b^{-1} \mathbf{H}_{bm}. \quad (45.9)$$

The matrix $\hat{\mathbf{H}}$ is known as the *generalized inertia matrix* for space manipulators [45.45]. This matrix represents the inertia property of the system in the joint space and can be mapped onto the Cartesian space using the generalized Jacobian matrix:

$$\hat{\mathbf{G}} = \hat{\mathbf{J}}\hat{\mathbf{H}}^{-1}\hat{\mathbf{J}}^T. \quad (45.10)$$

The matrix $\hat{\mathbf{G}}$ is referred to as the *inversed inertia tensor* for space manipulators and is useful for the discussion of impact dynamics when a space manipulator collides with or captures a floating target in orbit [45.46].

The *generalized jacobian matrix (GJM)* is a useful concept, with which the manipulator end point can perform positioning or trajectory tracking control by a simple control algorithm regardless of the attitude deviation during the operation.

A simplified laboratory demonstration was carried out using a two-dimensional free-floating test bed called *EFFORTS* [45.47]. To simulate the motion in a micro-gravity environment, a robot model was floated on a thin film of pressurized air on a horizontal plate, so that frictionless motion with momentum conservation was achieved.

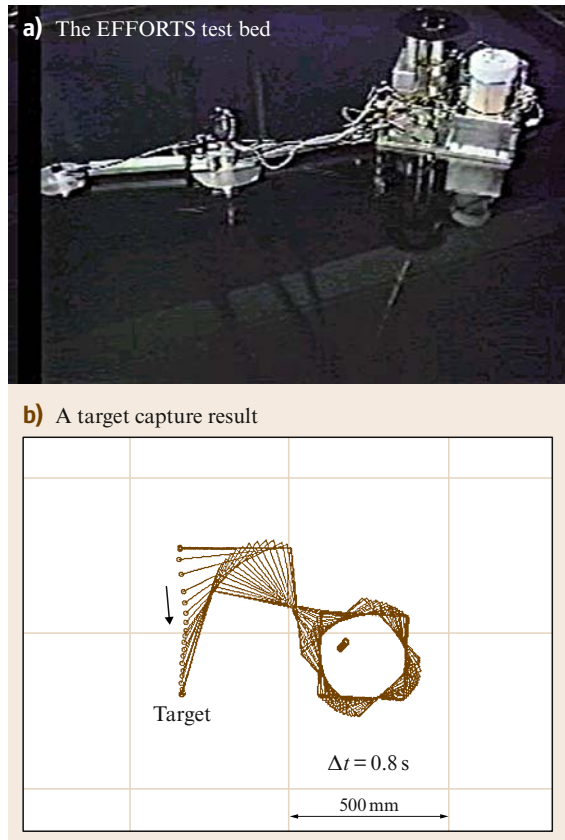


Fig. 45.20a,b Laboratory test bed for a free-floating space robot: (a) the EFFORTS test bed, (b) a target capture result

Figure 45.20 depicts the test bed and a typical experimental result. For Fig. 45.20b, the control command was given to the floating robot by

$$\dot{\phi} = \hat{J}^{-1} \dot{x}_d, \quad (45.11)$$

where \dot{x}_d is the desired velocity of the manipulator end point, the value of which was given and updated by online measurement of the end point position x_h and the target position x_t , as follows

$$\dot{x}_d = \frac{x_t - x_h}{\Delta t}, \quad (45.12)$$

where Δt is the time interval of the online control loop. The desired end point velocity was simply resolved into the joint velocity by (45.11).

The result clearly shows that the manipulator end point properly reached the target in an optimum manner, although the robot base rotated considerably due to the manipulator reaction. Note that, since the target was stationary in this example, the resulting motion trace was a straight line. However, thanks to the online control, the manipulator was also able to track and reach a moving target with the same control law.

The validity and effectiveness of the GJM-based manipulator control were also demonstrated in orbit by Japanese ETS-VII mission [45.16].

45.3.4 Linear and Angular Momenta

The integral of the upper set of equation in (45.1) gives a momentum equation, as shown in (45.4), which is composed of the linear and angular momenta. The linear part is expressed as

$$\tilde{H}_b v_b + \tilde{H}_{bm} \dot{\phi} = P, \quad (45.13)$$

where v_b is the linear velocity of the base, P is the initial linear momentum, and the inertia matrices with the over-cup are the corresponding components for the linear momentum [45.45]. When the linear momentum is further integrated, the result verifies the principle that the mass centroid of the entire system either remains stationary or translates with a constant velocity.

The angular momentum equation, however, does not have a second integral and therefore provides a first-order nonholonomic constraint [45.48]. The equation can be expressed as

$$\tilde{H}_b \omega_b + \tilde{H}_{bm} \dot{\phi} = L, \quad (45.14)$$

where ω_b is the angular velocity of the base, L is the initial angular momentum and the inertia matrices with the

over-tilde are the corresponding components for the angular momentum [45.45]. $\tilde{H}_{bm} \dot{\phi}$ represents the angular momentum generated by the manipulator motion.

Equation (45.14) can be solved for ω_b with zero initial angular momentum

$$\omega_b = -\tilde{H}_b^{-1} \tilde{H}_{bm} \dot{\phi}. \quad (45.15)$$

This expression describes the resulting disturbance motion of the base when there is joint motion $\dot{\phi}$ in the manipulator arm.

There are a number of points worth discussion when analyzing this equation. The magnitudes and directions of the maximum and minimum disturbances can be obtained from the singular value decomposition of the matrix $(-\tilde{H}_b^{-1} \tilde{H}_{bm})$ and displayed on the map. Such a map is called a *disturbance map* [45.49, 50]. Equation (45.15) is also used for the feedforward compensation in the coordinated manipulator–base control model [45.51, 52].

45.3.5 Virtual Manipulator

The concept of the *virtual manipulator (VM)* is an augmented kinematic representation that considers the base motion due to reaction forces or moments. The model is based on the fact that the mass centroid of the entire system does not move in the free-floating system without any external forces [45.53]. The mobility of the end point of the arm is decreased by the base motion. In the VM representation, such mobility degradation is expressed by virtually shrinking the length of the real arm according to the mass property. Note that the VM considers only conservation of linear momentum. If the differential expression of VM is obtained using a Jacobian matrix, the Jacobian is not a conventional kinematic Jacobian, but rather a version of the generalized Jacobian defined by the combination of the kinematic equation (45.2) and the linear momentum equation (45.13).

45.3.6 Dynamic Singularity

Dynamic singularities are singular configurations in which the manipulator end point loses mobility in some inertial direction [45.54]. Dynamic singularities are not found in Earth-based manipulators, but rather occur in free-floating space manipulator systems due to the coupling dynamics between the arm and the base. Dynamic singularities coincide with the singularities of the generalized Jacobian matrix determined by (45.6). The *singular value decomposition (SVD)*

of a manipulator Jacobian matrix provides *manipulability* analysis. Likewise, the SVD of the generalized Jacobian matrix yields the manipulability analysis of a free-floating space manipulator [45.55]. Figure 45.21 shows the comparison of the manipulability distribution between a two-degree-of-freedom (2-DOF) ground-based manipulator and a 2-DOF floating manipulator, from which the degradation of the manipulability is observed in the space arm due to the dynamic coupling.

45.3.7 Reaction Null Space (RNS)

From a practical point of view, any change in the base attitude is undesirable. As such, manipulator motion planning methods that minimize the base attitude disturbance have been investigated extensively. Analysis of the angular momentum equation reveals that the ultimate goal of achieving zero disturbance is possible.

The following is the angular momentum equation with zero initial angular momentum $\mathbf{L} = \mathbf{0}$ and the zero attitude disturbance $\omega_b = \mathbf{0}$ given in (45.14)

$$\tilde{\mathbf{H}}_{bm}\dot{\phi} = \mathbf{0}. \quad (45.16)$$

This equation yields the following null-space solution

$$\dot{\phi} = (\mathbf{E} - \tilde{\mathbf{H}}_{bm}^+ \tilde{\mathbf{H}}_{bm})\dot{\zeta}, \quad (45.17)$$

where \mathbf{E} is an identity matrix. The joint motion given by this equation is guaranteed not to disturb the base attitude. Here, the vector $\dot{\zeta} \in \mathbb{R}^n$ is arbitrary and the null space of the inertia matrix $\tilde{\mathbf{H}}_{bm} \in \mathbb{R}^{3 \times n}$ is called the *reaction null space (RNS)* [45.56].

The number of *degrees of freedom (DOF)* of $\dot{\zeta}$ is $n - 3$. For example, if a manipulator arm mounted on

a free-floating space robot has six DOFs, i.e., $n = 6$, then three DOFs remain in the reaction null space. These DOFs can be specified by introducing additional motion criteria, such as end point positioning of the arm. Such manipulator operation that produces no reaction in the base is called *reactionless manipulation* [45.57].

The validity and effectiveness of the RNS-based reactionless manipulation were demonstrated in orbit by Japanese ETS-VII mission [45.16].

45.3.8 Equations for Flexible-Based Manipulator Systems

Next, let us consider a flexible-based manipulator system in which a single or multiple manipulator arm(s) are supported by a flexible-beam or a spring and damper (viscoelastic) system. For such a system, the equation of motion is obtained using the following variables

- $\mathbf{x}_b \in \mathbb{R}^6$: position/orientation of the base
- $\phi \in \mathbb{R}^n$: joint angle of the arm
- $\mathbf{x}_h \in \mathbb{R}^{6k}$: position/orientation of the end point(s)
- $\mathbf{F}_b \in \mathbb{R}^6$: forces/moments to deflect the flexible base
- $\boldsymbol{\tau} \in \mathbb{R}^n$: joint torque of the arm
- $\mathbf{F}_h \in \mathbb{R}^{6k}$: external force/torque on the end point(s)

$$\begin{pmatrix} \mathbf{H}_b & \mathbf{H}_{bm} \\ \mathbf{H}_{bm}^\top & \mathbf{H}_m \end{pmatrix} \begin{pmatrix} \ddot{\mathbf{x}}_b \\ \ddot{\phi} \end{pmatrix} + \begin{pmatrix} \mathbf{c}_b \\ \mathbf{c}_m \end{pmatrix} = \begin{pmatrix} \mathbf{F}_b \\ \boldsymbol{\tau} \end{pmatrix} + \begin{pmatrix} \mathbf{J}_b^\top \\ \mathbf{J}_m^\top \end{pmatrix} \mathbf{F}_h \quad (45.18)$$

$$\dot{\mathbf{x}}_h = \mathbf{J}_m \dot{\phi} + \mathbf{J}_b \dot{\mathbf{x}}_b \quad (45.19)$$

$$\ddot{\mathbf{x}}_h = \mathbf{J}_m \ddot{\phi} + \dot{\mathbf{J}}_m \dot{\phi} + \mathbf{J}_b \ddot{\mathbf{x}}_b + \dot{\mathbf{J}}_b \dot{\mathbf{x}}_b. \quad (45.20)$$

Here, with the gravitational force \mathbf{g} in Cartesian space, the term \mathbf{c}_b is generally expressed as

$$\mathbf{c}_b = \mathbf{f}(\mathbf{x}_b, \phi, \dot{\mathbf{x}}_b, \dot{\phi}) + \mathbf{g}(\mathbf{x}_b, \phi). \quad (45.21)$$

The difference from the equation of a free-floating manipulator system (45.1) is the existence of the base constraint force \mathbf{F}_b . Let \mathbf{D}_b and \mathbf{K}_b be the damping and spring factors, respectively, of the flexible base. The constraint forces and moments \mathbf{F}_b are then expressed as

$$\mathbf{F}_b = -\mathbf{D}_b \dot{\mathbf{x}}_b - \mathbf{K}_b \Delta \mathbf{x}_b, \quad (45.22)$$

where $\Delta \mathbf{x}_b$ denotes the elastic base displacement from its neutral position.

Since the base is constrained, the total momentum is not conserved and it might be meaningless to check the system momentum. However, it is important to consider the partial momentum \mathcal{L}_m for the part of the manipulator arm:

$$\mathcal{L}_m = \mathbf{H}_{bm} \dot{\phi}, \quad (45.23)$$

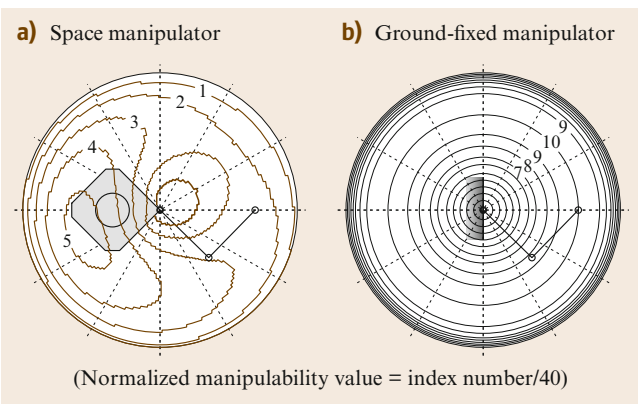


Fig. 45.21 Manipulability distribution in the work space

which is termed the *coupling momentum* [45.58]. Its time derivative describes the forces and moments \mathbf{F}_m , which are yielded by the dynamic reaction from the manipulator arm to the base

$$\mathbf{F}_m = \mathbf{H}_{bm}\ddot{\boldsymbol{\phi}} + \dot{\mathbf{H}}_{bm}\dot{\boldsymbol{\phi}}. \quad (45.24)$$

Using \mathbf{F}_b and \mathbf{F}_m , the upper set of equation in (45.18) is rearranged as

$$\mathbf{H}_b\ddot{\mathbf{x}}_b + \mathbf{D}_b\dot{\mathbf{x}}_b + \mathbf{K}_b\Delta\mathbf{x}_b = -\mathbf{g} - \mathbf{F}_m + \mathbf{J}_b^\top \mathbf{F}_h. \quad (45.25)$$

Equation (45.18) or (45.25), is a familiar expression for flexible-base manipulators [45.59, 60].

45.3.9 Advanced Control for Flexible Structure Based Manipulators

In this subsection, advanced control issues for a macro-micro space manipulator are discussed. The SSRMS attached to the SPDM (Fig. 45.4) and the JEMRMS (Fig. 45.6) are examples. Operation modes for this class of space manipulators include coarse motion by the macro (long-reach) component and fine manipulation by the micro component. In normal cases, these two control modes are executed exclusively. Namely, while one component is active the other component should be servo (or brake) locked. Thus, during the operation of the micro component, the macro component behaves just like a passive base.

Due to the flexible nature of the space long-reach arm, the macro part is subject to vibrations. These vibrations can be excited during coarse positioning and may remain for a long time after each operation. In fine manipulation, the macro arm behaves as a passive flexible structure, but then vibrations can be excited by reactions from the motion of the micro arm. These motions degrade the control accuracy and operational performance of the system. In practice, the booms are usually sufficiently stiff, but flexibility comes mainly from the low stiffness at the joints and gear trains. In addition, lightweight and microgravity characteristics make the structure sensitive to yield vibrations and the surrounding vacuum or the lack of air viscosity, provides a reduced damping effect to the structure.

Conventionally, the vibration issue has been managed by SRMS and SSRMS by the operational skill of well-trained astronauts and by limiting the maximum operational velocity according to the inertia of the handling object. However, if an advanced controller is introduced on the ISS, the training time for astronauts will be reduced and the operational speed can be increased.

Here, the following two subtasks are considered in dealing with this issue:

- suppression of the vibrations of the flexible base
- end point control in the presence of vibrations

To suppress the vibrations of the macro arm (flexible base), the coupled dynamics is effectively used. Such control is called *coupled vibration suppression control* [45.61]. The coupled dynamics is a solution space of the micro arm motion with maximum coupling with the vibration dynamics of the macro arm. Note that this solution space is perpendicular to the reaction null space introduced in Sect. 45.3.7. Since the spaces are orthogonal, the coupled vibration suppression control and reactionless manipulation can be superimposed without any mutual interference.

The motion command of the micro arm to suppress the vibrations is determined with a feedback of the linear and angular velocity of the end point of the macro arm $\dot{\mathbf{x}}_b$:

$$\ddot{\boldsymbol{\phi}} = \mathbf{H}_{bm}^+ \mathbf{H}_b \mathbf{G}_b \dot{\mathbf{x}}_b, \quad (45.26)$$

where $(\cdot)^+$ denotes the right pseudo-inverse and \mathbf{G}_b is a positive-definite gain matrix.

If written in the form of a joint torque input, the vibration control law is expressed as

$$\boldsymbol{\tau} = \mathbf{H}_m \mathbf{H}_{bm}^+ \mathbf{G}_b \dot{\mathbf{x}}_b. \quad (45.27)$$

In the presence of redundancy in the micro arm, (45.26) can be extended to control with a null-space component

$$\ddot{\boldsymbol{\phi}}_m = \mathbf{H}_{bm}^+ (\mathbf{H}_b \mathbf{G}_b \dot{\mathbf{x}}_b - \dot{\mathbf{H}}_{bm} \dot{\boldsymbol{\phi}}_m) + \mathbf{P}_{RNS} \boldsymbol{\zeta}, \quad (45.28)$$

where $\boldsymbol{\zeta}$ is an arbitrary n -DOF vector and $\mathbf{P}_{RNS} = (\mathbf{E} - \mathbf{H}_{bm}^+ \mathbf{H}_{bm})$ is a projector onto a null space of the coupling inertia matrix \mathbf{H}_{bm} . When the micro arm is operated using (45.28), the closed-loop system is expressed as

$$\mathbf{H}_b\ddot{\mathbf{x}}_b + \mathbf{H}_b \mathbf{G}_b \dot{\mathbf{x}}_b + \mathbf{K}_b \Delta\mathbf{x}_b = \mathbf{F}_b + \mathbf{J}_b^\top \mathbf{F}_h. \quad (45.29)$$

Equation (45.29) represents a second-order damped vibration system. With no force input, i. e., $\mathbf{F}_b = \mathbf{F}_h = 0$, the vibrations converge to zero with a proper choice of the gain matrix \mathbf{G}_b .

For the determination of the vector $\boldsymbol{\zeta}$, feedback control to reduce the positioning error of the micro arm end point is considered. The error vector is defined as

$$\tilde{\mathbf{x}}_h = \mathbf{x}_h^d - \mathbf{x}_h. \quad (45.30)$$

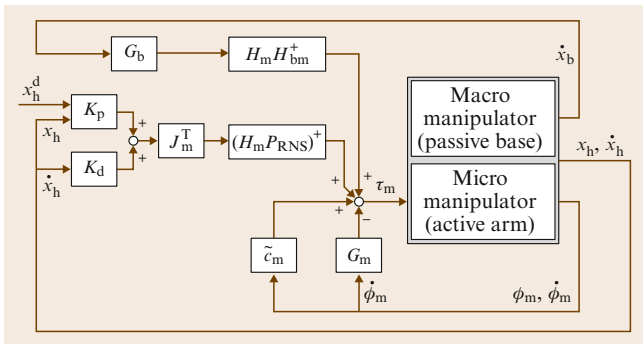


Fig. 45.22 Block diagram for simultaneous vibration suppression and manipulator end point control for a flexible-structure-mounted manipulator system

After some manipulation, the control law for the joint torque of the micro arm is obtained in the form [45.61, 62]

$$\tau = [(J_m^T)^+ H_m P_{RNS}]^+ (K_p \tilde{x}_h - K_d J_m \dot{\phi}) - G_m \dot{\phi}, \quad (45.31)$$

where K_p , K_d , and G_m are positive-definite gain matrices.

Figure 45.22 shows a block diagram for the control system described by Eqs. (45.27) and (45.31).

As a simplified demonstration, a planar system with a four-joint redundant manipulator arm atop a flexible beam is considered. Figure 45.23 shows the vibration amplitude of the flexible beam after an impulsive external force. The trace labeled 'w/o vs' depicts the vibrations of the beam without any manipulator control but with natural damping. The graph labeled 'with vs' depicts the case in which vibration suppression control given by (45.27) is applied, where the vibrations are damped quickly.

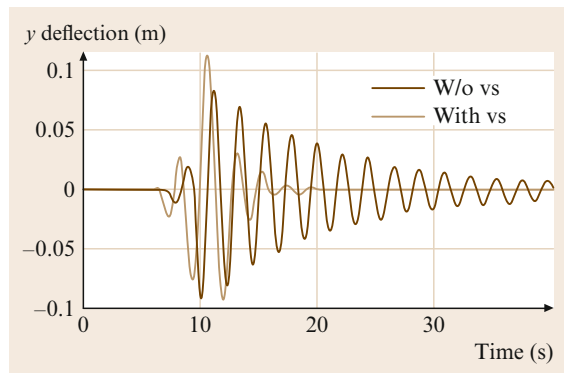


Fig. 45.23 Vibrations of the flexible base

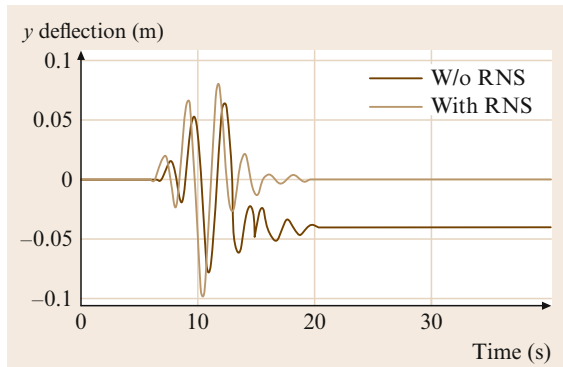


Fig. 45.24 Positioning error of the manipulator end tip

In addition, Fig. 45.24 shows the end point motion of the manipulator arm during the control. The graph labeled 'w/o RNS' is the case of using (45.27), where the base vibrations were successfully suppressed but the position of the manipulator end point was deflected by this suppression behavior. The graph labeled 'with RNS' depicts the case in which both vibration suppression control given by (45.27) and the end point control given by (45.31) are applied simultaneously. This last case shows that the vibrations were damped successfully and that the positioning error of the manipulator end point converged to zero. This is a result of the redundancy of the arm.

Here, note that the proposed control method requires precise information on dynamic characteristics, such as the inertia parameters of the arms and the handled payload, if any. To achieve more practical applications, the proposed method must be extended to schemes for parameter identification [45.63] and adaptive control [45.64], with which the convergence of the control is guaranteed even with imprecise a priori knowledge of the dynamic parameters [45.62].

45.3.10 Contact Dynamics and Impedance Control

The capture and retrieval operation of a floating and tumbling *target*, such as a malfunctioning satellite, by a manipulator arm mounted on a servicing robot (called a *chaser*) can be decomposed into the following three phases:

1. approach phase (before contact with the target)
2. contact/impact phase (at the moment of contact)
3. post-contact phase (after contact or grasping)

If the contact is impulsive, the first and the third phases are discontinued by the impulsive phenomena of the second phase. Understanding of this impulsive phenomena is indispensable when designing a comprehensive capture control scheme. In this subsection, the formulation of impact dynamics is first considered and impedance control (which is useful to minimize the impact forces and prolong the contact duration) is then discussed.

Let us consider a chain of rigid links composed of $n + 1$ bodies freely floating in inertial space. As discussed in Sect. 45.3.3, the equation of motion for this type of system becomes (45.8). Here, the impulsive contact force is assumed to yield at the manipulator end tip and is expressed as $\mathbf{F}_h = (\mathbf{f}_h^\top, \mathbf{n}_h^\top)^\top \in \mathbb{R}^6$. This impulsive force also yields a change in the system momenta $(\mathbf{P}_g^\top, \mathbf{L}_g^\top)^\top \in \mathbb{R}^6$, as expressed by

$$\begin{aligned} \begin{pmatrix} \dot{\mathbf{P}}_g \\ \dot{\mathbf{L}}_g \end{pmatrix} &= \begin{pmatrix} w\mathbf{E} & 0 \\ 0 & \mathbf{I}_g \end{pmatrix} \begin{pmatrix} \dot{\mathbf{v}}_g \\ \dot{\boldsymbol{\omega}}_g \end{pmatrix} + \begin{pmatrix} 0 \\ \boldsymbol{\omega}_g \times \mathbf{I}_g \boldsymbol{\omega}_g \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{E} & 0 \\ \tilde{\mathbf{r}}_{gh} & \mathbf{E} \end{pmatrix} \begin{pmatrix} \mathbf{f}_h \\ \mathbf{n}_h \end{pmatrix}, \end{aligned} \quad (45.32)$$

where \mathbf{E} is a 3×3 identity matrix, w is a total mass, \mathbf{I} is an inertia matrix, and the symbols with the suffix ‘g’ indicate the corresponding values observed around the mass centroid of the $(n + 1)$ -link system.

In the above equations, (45.8) describes an internal joint motion (termed *local motion*) of the system, whereas (45.32) describes the overall motion (termed *global motion*) about the centroid of the system. As a result of force input \mathbf{F}_h , the floating chain induces both the local motion around its articulated joints and the global translation and rotation with respect to the centroid.

From (45.8) and (45.32), the acceleration of the manipulator end tip $\boldsymbol{\alpha}_h$ can be expressed in the inertial frame as

$$\boldsymbol{\alpha}_h = \mathbf{G}^* \mathbf{F}_h + \mathbf{d}^*, \quad (45.33)$$

where

$$\mathbf{G}^* = \hat{\mathbf{J}} \hat{\mathbf{H}}^{-1} \hat{\mathbf{J}}^\top + \mathbf{R}_h \mathbf{M}^{-1} \mathbf{R}_h^\top, \quad (45.34)$$

$$\mathbf{R}_h = \begin{pmatrix} \mathbf{E} & -\tilde{\mathbf{r}}_{gh} \\ 0 & \mathbf{E} \end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} w\mathbf{E} & 0 \\ 0 & \mathbf{I}_g \end{pmatrix}, \quad (45.35)$$

and \mathbf{d}^* is a velocity-dependent term.

Equations (45.33)–(45.35) are expressions for the motion of the hand (the point at which collision occurs) induced by the impact force \mathbf{F}_h , where the matrix \mathbf{G}^* , which is the augmented version of (45.10), represents the dynamic characteristics of this system.

Further augmentation for the inverted inertia matrix has been discussed for the case in which the contact duration is not considered to be infinitesimal [45.46].

Now let us assume the case in which two free-floating chains, A and B, with dynamic characteristics \mathbf{G}_A^* and \mathbf{G}_B^* collide with each other at their respective hands (end points) and an impact force \mathbf{F}_h is induced by this collision.

The equations of motion at the instance of collision are

$$\mathbf{G}_A^* \mathbf{F}_h = \begin{pmatrix} \dot{\mathbf{v}}_{hA} \\ \dot{\boldsymbol{\omega}}_{hA} \end{pmatrix} - \mathbf{d}_A^* \quad (45.36)$$

for chain A and

$$\mathbf{G}_B^* (-\mathbf{F}_h) = \begin{pmatrix} \dot{\mathbf{v}}_{hB} \\ \dot{\boldsymbol{\omega}}_{hB} \end{pmatrix} - \mathbf{d}_B^* \quad (45.37)$$

for chain B, where the subscripts A and B indicate the label of the chain.

Assuming that \mathbf{G}_A^* and \mathbf{G}_B^* remain constant during the infinitesimal contact duration and that the velocity-dependent terms \mathbf{d}_A^* and \mathbf{d}_B^* are small and negligible, integration of (45.36) and (45.37) yields

$$\mathbf{G}_A^* \int_t^{t+\delta t} \mathbf{F}_h dt = \begin{pmatrix} \mathbf{v}'_{hA} \\ \boldsymbol{\omega}'_{hA} \end{pmatrix} - \begin{pmatrix} \mathbf{v}_{hA} \\ \boldsymbol{\omega}_{hA} \end{pmatrix}, \quad (45.38)$$

$$\mathbf{G}_B^* \int_t^{t+\delta t} \mathbf{F}_h dt = \begin{pmatrix} \mathbf{v}_{hB} \\ \boldsymbol{\omega}_{hB} \end{pmatrix} - \begin{pmatrix} \mathbf{v}'_{hB} \\ \boldsymbol{\omega}'_{hB} \end{pmatrix}, \quad (45.39)$$

where $\{ '\}$ indicates the velocity after the collision. Integration of \mathbf{F}_h

$$\bar{\mathbf{F}}_h = \lim_{\delta t \rightarrow 0} \int_t^{t+\delta t} \mathbf{F}_h dt \quad (45.40)$$

represents the impulse (force-time product) acting on both chains. Providing that the total momenta of the two systems are strictly conserved before and after the collision, we obtain the following expression from (45.38) and (45.39).

$$\begin{aligned} (\mathbf{G}_A^* + \mathbf{G}_B^*) \bar{\mathbf{F}}_h &= \left[\begin{pmatrix} \mathbf{v}'_{hA} \\ \boldsymbol{\omega}'_{hA} \end{pmatrix} - \begin{pmatrix} \mathbf{v}'_{hB} \\ \boldsymbol{\omega}'_{hB} \end{pmatrix} \right] \\ &\quad + \left[\begin{pmatrix} \mathbf{v}_{hB} \\ \boldsymbol{\omega}_{hB} \end{pmatrix} - \begin{pmatrix} \mathbf{v}_{hA} \\ \boldsymbol{\omega}_{hA} \end{pmatrix} \right]. \end{aligned} \quad (45.41)$$

In general collision analysis, the coefficient of restitution (elasticity factor) associated with the relative

velocities before and after the collision is often employed [45.65]. If we accept this restitution coefficient for six-DOF linear and angular velocities, the relationship between relative velocities before and after contact becomes

$$\begin{pmatrix} v'_{hA} \\ \omega'_{hA} \end{pmatrix} - \begin{pmatrix} v'_{hB} \\ \omega'_{hB} \end{pmatrix} = \epsilon \left[\begin{pmatrix} v_{hB} \\ \omega_{hB} \end{pmatrix} - \begin{pmatrix} v_{hA} \\ \omega_{hA} \end{pmatrix} \right], \quad (45.42)$$

where

$$\epsilon \equiv \text{diag}(\epsilon_1, \dots, \epsilon_6) \quad 0 \leq \epsilon_i \leq 1 \quad (45.43)$$

is the restitution coefficient matrix.

Substituting (45.42) into (45.41), the impulse induced by this collision can be expressed only by the relative velocity of two points before the contact

$$\bar{F}_h = (E_6 + \epsilon) G_\Sigma^{*-1} \mathcal{V}_{hAB}, \quad (45.44)$$

where

$$G_\Sigma^* = G_A^* + G_B^* \quad (45.45)$$

$$\mathcal{V}_{hAB} = \begin{pmatrix} v_{hB} \\ \omega_{hB} \end{pmatrix} - \begin{pmatrix} v_{hA} \\ \omega_{hA} \end{pmatrix}. \quad (45.46)$$

Using the above notation, the magnitude of impulse is expressed as

$$\|\bar{F}_h\| = \sqrt{(E_6 + \epsilon)^T \mathcal{V}_{hAB}^T G_\Sigma^{*-T} G_\Sigma^{*-1} \mathcal{V}_{hAB} (E_6 + \epsilon)} \quad (45.47)$$

and the velocity after collision becomes

$$\begin{pmatrix} v'_{hA} \\ \omega'_{hA} \end{pmatrix} = (G_A^{*-1} + G_B^{*-1})^{-1} \left[(E_6 + \epsilon) G_B^{*-1} \begin{pmatrix} v_{hB} \\ \omega_{hB} \end{pmatrix} + (G_A^{*-1} - \epsilon G_B^{*-1}) \begin{pmatrix} v_{hA} \\ \omega_{hA} \end{pmatrix} \right], \quad (45.48)$$

where the suffixes A and B are interchangeable.

These expressions are considered to be an augmentation of the impact theory for a two-point-mass system into articulated body systems.

Impedance control is a concept by which to control the manipulator end tip so as to obtain the desired mechanical impedance characteristics. Such control is useful to alter the dynamic characteristics of the arm during the contact phase. In a special case, the desired impedance of the manipulator end tip (hand) may be tuned to achieve *impedance matching* with the colliding target object so that the hand can easily maintain stable contact with the target [45.66].

Let M_d , D_d , and K_d be matrices for the desired impedance properties of inertia, viscosity, and stiffness, respectively, measured at the manipulator end point. The equation of motion for the desired system is then expressed as

$$M_d \ddot{x}_h + D_d \dot{\Delta x}_h + K_d \Delta x_h = F_h. \quad (45.49)$$

From (45.8) and (45.49), the impedance control law for a free-floating manipulator system is obtained as follows [45.67]:

$$\tau_h = H^* \hat{J}^{-1} [M_d^{-1} (D_d \dot{\Delta x}_h + K_d \Delta x_h - F_h) - \dot{J} \dot{\phi} - \ddot{x}_{gh}] - \hat{J}^T F_h + c^*. \quad (45.50)$$

The usefulness of the impedance control in free-flying space robots has been discussed in [45.67–69].

45.3.11 Dynamics of Mobile Robots

The equation of motion for a mobile robot that has multiple articulated limbs, such as that shown in Fig. 45.25, is given by the following equation

$$\begin{pmatrix} H_b & H_{bm1} & \cdots & H_{bmk} \\ H_{bm1}^T & H_{m11} & \cdots & H_{m1k} \\ \vdots & \vdots & \ddots & \vdots \\ H_{bmk}^T & H_{m1k}^T & \cdots & H_{mkk} \end{pmatrix} \begin{pmatrix} \ddot{x}_b \\ \ddot{\phi}_1 \\ \vdots \\ \ddot{\phi}_k \end{pmatrix} + \begin{pmatrix} c_b \\ c_{m1} \\ \vdots \\ c_{mk} \end{pmatrix} = \begin{pmatrix} F_b \\ \tau_1 \\ \vdots \\ \tau_n \end{pmatrix} + \begin{pmatrix} J_b^T F_{ex} \\ J_{m1}^T F_{ex1} \\ \vdots \\ J_{mk}^T F_{exk} \end{pmatrix}, \quad (45.51)$$

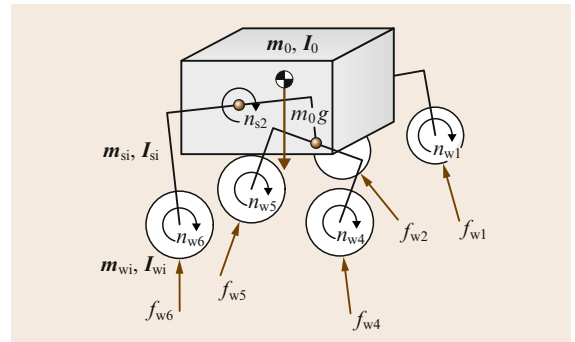


Fig. 45.25 Schematic model of a mobile robot

where the symbols have the following meanings

k :	number of limbs
$\mathbf{x}_b \in \mathbb{R}^6$:	position/orientation of the base
$\boldsymbol{\phi} = (\boldsymbol{\phi}_1^\top, \dots, \boldsymbol{\phi}_k^\top)^\top \in \mathbb{R}^n$:	articulated joint angles
$\mathbf{x}_{ex} = (\mathbf{x}_{ex1}^\top, \dots, \mathbf{x}_{exk}^\top)^\top \in \mathbb{R}^{6k}$:	position/orientation of the end points
$\mathbf{F}_b \in \mathbb{R}^6$:	forces/moments directly apply on the base
$\boldsymbol{\tau} = (\boldsymbol{\tau}_1^\top, \dots, \boldsymbol{\tau}_k^\top)^\top \in \mathbb{R}^n$:	joint articulated torque
$\mathbf{F}_{ex} = (\mathbf{F}_{ex1}^\top, \dots, \mathbf{F}_{exk}^\top)^\top \in \mathbb{R}^{6k}$:	external forces/moments on the end points

Note that for Fig. 45.25, $\mathbf{F}_{exi} = (\mathbf{f}_{wi}^\top, \mathbf{n}_{wi}^\top)^\top$.

Comparing the above equations with (45.1), no difference is observed in the mathematical structure. The gravity force on the vehicle main body and the configuration-dependent gravity terms of the articulated bodies are included in \mathbf{c}_b and \mathbf{c}_{mi} , respectively. In practice, however, one substantial difference is the existence of ground contact forces/moments at the end point of each limb. Unlike floating target capture discussed in Sect. 45.3.10, the contact is not considered impulsive, but instead continues for a nonnegligible period of time. In such cases, a well-accepted approach is to evaluate the contact forces/moments \mathbf{F}_{ex} explicitly, according to the virtual penetration of the end point into the collided object or the ground surface [45.65].

In cases in which each limb has a wheel on its end terminal, rather than the point penetration model, a wheel traction model shall be adopted to evaluate \mathbf{F}_{ex} . For planetary exploration missions, rovers (mobile robots) are expected to travel over natural rough terrain. A number of studies have examined the modeling of tire traction forces on loose soil, called *regolith* (where there is no organic component) [45.70–78].

In the following subsection, models for wheel traction mechanics are summarized.

45.3.12 Wheel Traction Mechanics

Terramechanics is the study of soil properties, specifically the interaction of wheeled, legged, or tracked vehicles and various surfaces. For the modeling of the wheel traction forces and the analysis of the vehicle mobility, textbooks written by *Bekker* [45.25, 26] and *Wong* [45.79] are good references. Although these books

were written in the 1960s and 1970s, basic formulae from these books are frequently cited by researchers even today [45.80]. In this subsection, the models for a rigid wheel on loose soil are summarized.

Slip Ratio and Slip Angle

Slips are generally observed when a rover travels on loose soil. In particular, slips in the lateral direction are observed during steering or slope-traversing maneuvers.

The slip in the longitudinal direction is measured by the slip ratio s , which is defined as a function of the longitudinal traveling velocity v_x of the wheel and the circumference velocity of the wheel $r\omega$ (r is the wheel radius and ω is the angular velocity of the wheel):

$$s = \begin{cases} (r\omega - v_x)/r\omega, & \text{if } |r\omega| > |v_x| : \text{driving}, \\ (r\omega - v_x)/v_x, & \text{if } |r\omega| < |v_x| : \text{braking}. \end{cases} \quad (45.52)$$

The slip ratio takes a value of from -1 to 1 .

On the other hand, the slip in the lateral direction is measured by the slip angle β , which is defined in terms of v_x and the lateral traveling velocity v_y as

$$\beta = \tan^{-1}(v_y/v_x). \quad (45.53)$$

Note that the above definitions, Eqs. (45.52) and (45.53), have traditionally been used in the vehicle community as standards. However, planetary rovers in a challenging terrain, such as Spirit and Opportunity on Mars, experienced the cases in which the rovers slip backward while attempting to drive up hill or travel faster than the wheel's circumference velocity in downhill driving. In these cases the slip ratio can exceed the range from -1 to 1 . Also while traversing side slopes, the case may arise in which $v_y > 0$ but v_x is nearly 0, making the definition (45.53) nearly singular. Therefore,

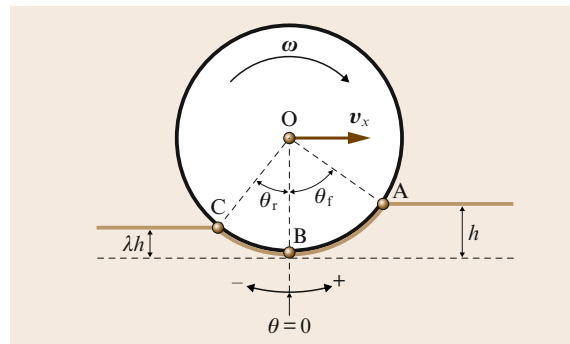


Fig. 45.26 Wheel contact angles

these definitions need to be further discussed for rovers in very loose terrain.

Wheel–Soil Contact Angle

Figure 45.26 depicts a schematic model of a rigid wheel contacting loose soil. In the figure, the angle from the surface normal to the point at which the wheel initially makes contact with the soil ($\angle AOB$) is defined as the entry angle. The angle from the surface normal to the point at which the wheel departs from the soil ($\angle BOC$ in Fig. 45.26) is the exit angle. The wheel contact region on loose soil is represented from the entry angle to the exit angle.

The entry angle θ_f is geometrically described in terms of wheel sinkage h as

$$\theta_f = \cos^{-1}(1 - h/r). \quad (45.54)$$

The exit angle θ_r is described using the wheel sinkage ratio λ , which denotes the ratio between the forward and rear sinkage of the wheel

$$\theta_r = \cos^{-1}(1 - \lambda h/r). \quad (45.55)$$

The value of λ depends on the soil characteristics, the wheel surface pattern and the slip ratio. It becomes

smaller than 1.0 when the soil compaction occurs, but can be greater than 1.0 when the soil is dug up by the wheel and transported to the rear region of the wheel.

Wheel Sinkage

The amount of wheel sinkage is constituted by static and dynamic components. The static sinkage depends on the vertical load on the wheel, while the dynamic sinkage is caused by the rotation of the wheel.

According to the equation formulated by Bekker [45.25], the static stress $p(h)$ generated under a flat plate, which has a sinkage h and a width b , is calculated as

$$p(h) = (k_c/b + k_\phi)h^n, \quad (45.56)$$

where k_c and k_ϕ are pressure-sinkage modules and n is the sinkage exponent. Applying (45.56) to the wheel, as shown in Fig. 45.27, the static sinkage is evaluated as follows.

First, the wheel sinkage $h(\theta)$ at an arbitrary wheel angle θ is geometrically given by

$$h(\theta) = r(\cos \theta - \cos \theta_s), \quad (45.57)$$

where θ_s is the static contact angle. Then, substituting (45.57) into (45.56) yields

$$p(\theta) = r^n(k_c/b + k_\phi)(\cos \theta - \cos \theta_s)^n. \quad (45.58)$$

The wheel eventually sinks into the soil until the stress from the soil balances the vertical load W on the wheel.

$$\begin{aligned} W &= \int_{-\theta_s}^{\theta_s} p(\theta) b r \cos \theta d\theta \\ &= r^{n+1} (k_c + k_\phi b) \int_{-\theta_s}^{\theta_s} (\cos \theta - \cos \theta_s)^n \cos \theta d\theta. \end{aligned} \quad (45.59)$$

Using this equation, the static contact angle θ_s is evaluated for the given W . In practice, (45.59) does not yield a closed-form solution for θ_s , although θ_s can be evaluated numerically.

Finally, the static sinkage h_s is obtained by substituting θ_s into the following equation

$$h_s = r(1 - \cos \theta_s). \quad (45.60)$$

However, as illustrated in Fig. 45.28, the dynamic sinkage becomes a complicated function depending on the slip ratio of the wheel, the wheel surface pattern, and the soil characteristics. Although it is difficult to obtain an analytical form for the dynamic sinkage, it is again

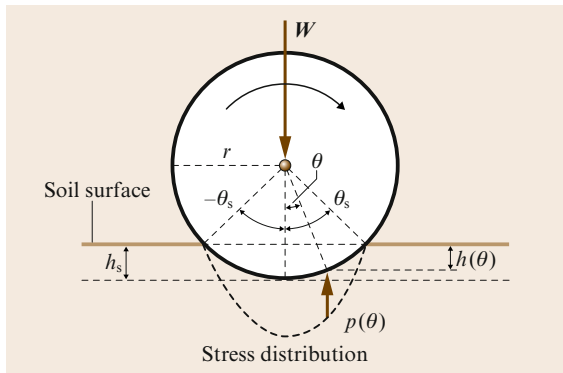


Fig. 45.27 Static sinkage

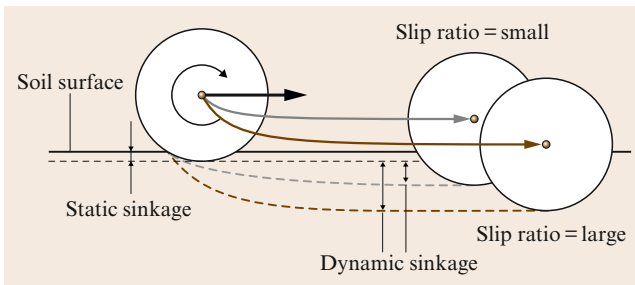


Fig. 45.28 Dynamic sinkage

possible to evaluate the dynamic sinkage numerically, using the condition $W = F_z$, where F_z is the normal force given by (45.71), which will be presented later herein. The force F_z increases with the wheel sinkage because the area of the contact patch increases accordingly.

Stress Distribution under the Wheel

Based on terramechanics models, the stress distribution under the rotating wheel can be modeled as shown in Fig. 45.29.

The normal stress $\sigma(\theta)$ is determined by the following equation [45.80, 81]:

$$\sigma(\theta) = r^n \left(\frac{k_c}{b} + k_\phi \right) [\cos \theta - \cos \theta_f]^n \quad \text{for } \theta_m \leq \theta < \theta_f$$

$$\sigma(\theta) = r^n \left(\frac{k_c}{b} + k_\phi \right) \left\{ \cos \left[\theta_f - \frac{\theta - \theta_r}{\theta_m - \theta_r} (\theta_f - \theta_m) \right] - \cos \theta_f \right\}^n \quad \text{for } \theta_r < \theta \leq \theta_m \quad (45.61)$$

Note that the above equations are based on Bekker's formula, as given in (45.56) and they become equivalent to the Wong–Reece model for normal stress [45.82] when $n = 1$. Note also that, by linearizing this distribution, Iagnemma et al. [45.70, 80] developed a Kalman-filter-based method to estimate the soil parameters.

The term θ_m is the specific wheel angle at which the normal stress is maximized

$$\theta_m = (a_0 + a_1 s) \theta_f, \quad (45.62)$$

where a_0 and a_1 are parameters that depend on the wheel–soil interaction. Their values are generally assumed to be $a_0 \approx 0.4$ and $0 \leq a_1 \leq 0.3$ [45.82].

The maximum terrain shear force is a function of the terrain cohesion c and internal friction angle ϕ and can be computed from Coulomb's equation

$$\tau_{\max}(\theta) = c + \sigma_{\max}(\theta) \tan \phi. \quad (45.63)$$

Based on the above equation, the shear stresses under the rotating wheel, $\tau_x(\theta)$ and $\tau_y(\theta)$, are written as [45.83]

$$\tau_x(\theta) = (c + \sigma(\theta) \tan \phi) (1 - e^{-j_x(\theta)/k_x}), \quad (45.64)$$

$$\tau_y(\theta) = (c + \sigma(\theta) \tan \phi) (1 - e^{-j_y(\theta)/k_y}), \quad (45.65)$$

where k_x and k_y are the shear deformation moduli in each direction. In addition, j_x and j_y , which are the

soil deformations in each direction, can be formulated as a function of the wheel angle θ with the slip ratio and the slip angle, respectively [45.73, 82]

$$j_x(\theta) = r[\theta_f - \theta - (1 - s)(\sin \theta_f - \sin \theta)], \quad (45.66)$$

$$j_y(\theta) = r(1 - s)(\theta_f - \theta) \tan \beta. \quad (45.67)$$

Drawbar Pull: F_x

Using the normal stress $\sigma(\theta)$ and the shear stress in the x direction $\tau_x(\theta)$, the drawbar pull F_x , which is the net traction force exerted from the soil to the wheel, is calculated as the integral from the entry angle θ_r to the exit angle θ_f [45.82]

$$F_x = rb \int_{\theta_r}^{\theta_f} \{ \tau_x(\theta) \cos \theta - \sigma(\theta) \sin \theta \} d\theta. \quad (45.68)$$

Side Force: F_y

The side force F_y appears in the lateral direction of the wheel when the vehicle makes steering maneuvers or traverses a side slope. The side force is decomposed into two components [45.63]

$$F_y = F_u + F_s,$$

where F_u is the force produced by the shear stress in the y direction $\tau_y(\theta)$ underneath the wheel and F_s is the reaction force generated by the bulldozing phenomenon on a side face of the wheel. The above equation can be

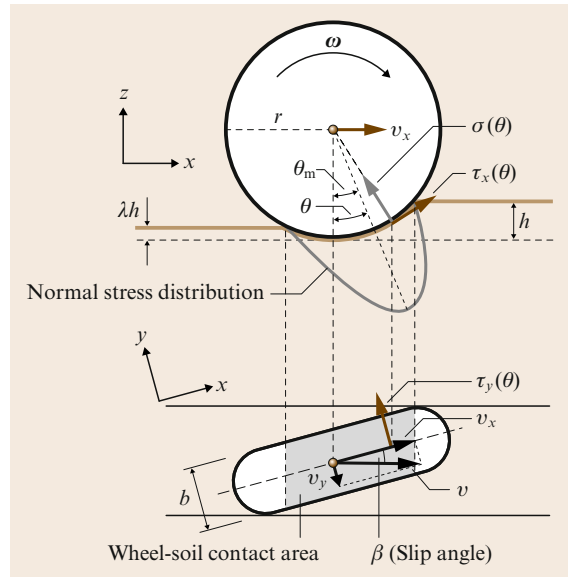


Fig. 45.29 Stress distribution model under a wheel

rewritten as

$$F_y = \int_{\theta_r}^{\theta_f} \underbrace{\{rb\tau_y(\theta)\}}_{F_u} + \underbrace{R_b[r - h(\theta)\cos\theta]}_{F_s} d\theta. \quad (45.69)$$

Here, *Hegedus's bulldozing resistance estimation* [45.84] is employed to evaluate the side face force F_s . As shown in Fig. 45.30, a bulldozing resistance R_b is generated on a unit width blade when the blade moves towards the soil. According to Hegedus's theory, the bulldozed area is defined by a destructive phase that is modeled by a planar surface. In the case of a horizontally placed wheel, the angle of approach α' should be zero. R_b can then be calculated as a function of wheel sinkage $h(\theta)$:

$$R_b(h) = D_1 \left[ch(\theta) + D_2 \frac{\rho_d h^2(\theta)}{2} \right], \quad (45.70)$$

where

$$D_1(X_c, \phi) = \cot X_c + \tan(X_c + \phi),$$

$$D_2(X_c, \phi) = \cot X_c + \cot^2 X_c / \cot \phi.$$

In the above equations, ρ_d denotes the soil density. Based on Bekker's theory [45.25], the destructive angle X_c can be approximated as

$$X_c = 45^\circ - \phi/2.$$

Normal Force: F_z

The normal force F_z is obtained in the same manner as for (45.68) [45.82]:

$$F_z = rb \int_{\theta_r}^{\theta_f} [\tau_x(\theta) \sin \theta + \sigma(\theta) \cos \theta] d\theta, \quad (45.71)$$

which should balance the normal load of the wheel in a static condition.

Motion dynamics simulation for a vehicle traveling over loose soil can be performed by substituting the forces F_x , F_y , and F_z obtained from the above equations into the equation of motion of (45.51).

A better understanding of the soil-wheel contact and traction mechanics is important in order to

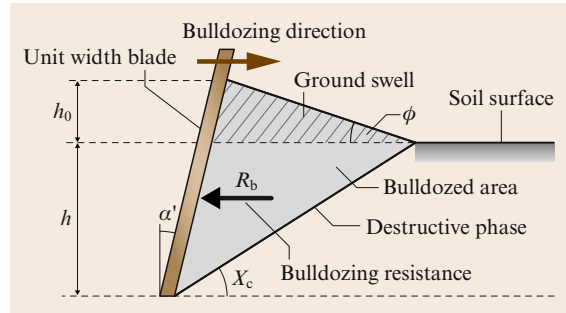


Fig. 45.30 Estimation model of the bulldozing resistance

improve the navigation and control behavior of exploration rovers, in terms of minimization of wheel slippage, for example. Reducing the wheel slippage will increase the power efficiency of surface locomotion, decrease the errors in path tracking maneuvers and decrease the risks of wheel spinning and sinking, which can cause immobilization of the vehicle.

One key to realizing such advanced control of slippage minimization is determining how to properly estimate the slip ratios and slip angles in real time using onboard sensors. The slip ratio is determined by the ratio between the wheel spinning velocity and the traveling velocity of the vehicle, but proper sensing of the velocity of the vehicle is usually difficult. One simple solution is to use a free wheel specialized for traveling velocity measurement. Another solution is to employ inertial sensors, which are however usually subject to noise and drift.

An alternative, but promising, possibility is visual odometry, which is based on optical flow or feature tracking in the sequence of optical images. Actually, this technique has been applied to the Mars exploration rovers, Spirit and Opportunity, in their long-range navigation, and there verified to be very useful. Particularly, the algorithm based on feature detection and tracking using a stereo pair of cameras provides reliable results with good accuracy for the estimation of driving distance as well as the wheel slippage [45.24].

45.4 Future Directions of Orbital and Surface Robotic Systems

45.4.1 Robotic Maintenance and Service Missions

Robotic maintenance and service missions for space infrastructure have been a long-term dream in the

space robotics community since their conceptual designs were first published in ARAMIS report in early 1980s (Fig. 45.1) [45.1].

ROTEX, ETS-VII, Ranger, and ASTRO, which were introduced in earlier sections, are technological devel-



Fig. 45.31 A conceptual drawing for robotic rescue of Hubble space telescope

opments toward this goal, but robotic maintenance and service missions have not become routinely operational yet. A good comparative study of orbital robotic missions is provided by [45.85]. The *Hubble space telescope* (HST) is a huge space telescope which has the capability to be serviced in orbit, but it has been visited by Space Shuttle and serviced (components exchanged and problems fixed) only by human EVA. After the COLUMBIA accident in 2003, NASA seriously considered the possibility of robotic maintenance of the HST, investigating available technologies and selecting a prime contractor of the mission development. Figure 45.31 depicts one possible configuration for the robotic rescue mission. Ultimately, it was decided to perform this last servicing mission with human astronauts. Maintenance of the HST involves tasks that are currently too complicated to be done by a robot, because the HST itself was designed for human-based maintenance and not specifically designed for robots.

Robonaut, which is described in the following subsection, is therefore considered as an interesting option for conducting practical maintenance and service missions due to its compatibility and similar level of dexterity as human astronauts.

45.4.2 Robonaut

Robonaut (Fig. 45.32a) is a humanoid robot designed by the Robot Systems Technology Branch at the NASA Johnson Space Center in a collaborative effort with DARPA. The Robonaut project seeks to develop and demonstrate a robotic system that can function as an EVA astronaut equivalent. Robonaut jumps generations ahead by eliminating the robotic scars (e.g., special

robotic grapples and targets), but it still keeps the human operator in the control loop through its telepresence control system. Robonaut is designed to be used for EVA tasks not specifically designed for robots [45.86].

A key challenge is to build machines that can help humans work and explore in space. Working side by side with humans or going where the risks are too great for people, machines like Robonaut will expand capabilities for construction and discovery. Over the past five decades, space flight hardware has been designed for human servicing. Space walks are planned for most of the assembly missions for the International Space Station and they are a key contingency for resolving on-orbit failures. To maintain compatibility with existing EVA tools and equipments, a humanoid shape and an assumed level of human performance (at least a human in a space suit) are required for this robotic surrogate.

The manipulator and dexterous hand have been developed with a substantial investment in mechatronics design. The arm structure has embedded avionics elements within each link, reducing cabling and noise interference. Robonaut has been designed based on a biologically inspired approach. For example, it uses a chordate neurological system in data management, bringing all feedback to a central nervous system, where even low-level servo control is performed. Such a biologically inspired approach is extended to left-right computational symmetry, sensor and power duality and kinematical redundancy, enabling learning and optimization in mechanical, and electrical and software forms.

Robonaut has a broad mix of sensors including thermal, position, tactile, force, and torque instrumentation, with over 150 sensors per arm. The control system for Robonaut includes an onboard, real-time central processing unit (CPU) with miniature data acquisition and power management. Off-board guidance is delivered with human supervision using a telepresence control station with human tracking.

One potential application of Robonaut is a regular monitoring and contingent maintenance work of human habitation modules of the space station. Fig. 45.32b, depicts such an application where Robonaut crawls on the surface of the station module by using hand rails which were originally designed for human EVA.

The application of Robonaut technology is not limited to orbital tasks. Figure 45.32c,d depicts conceptual designs at NASA and DLR, respectively, to combine the humanoid torso on a surface mobility system, which will be useful for robotic planetary explorations.

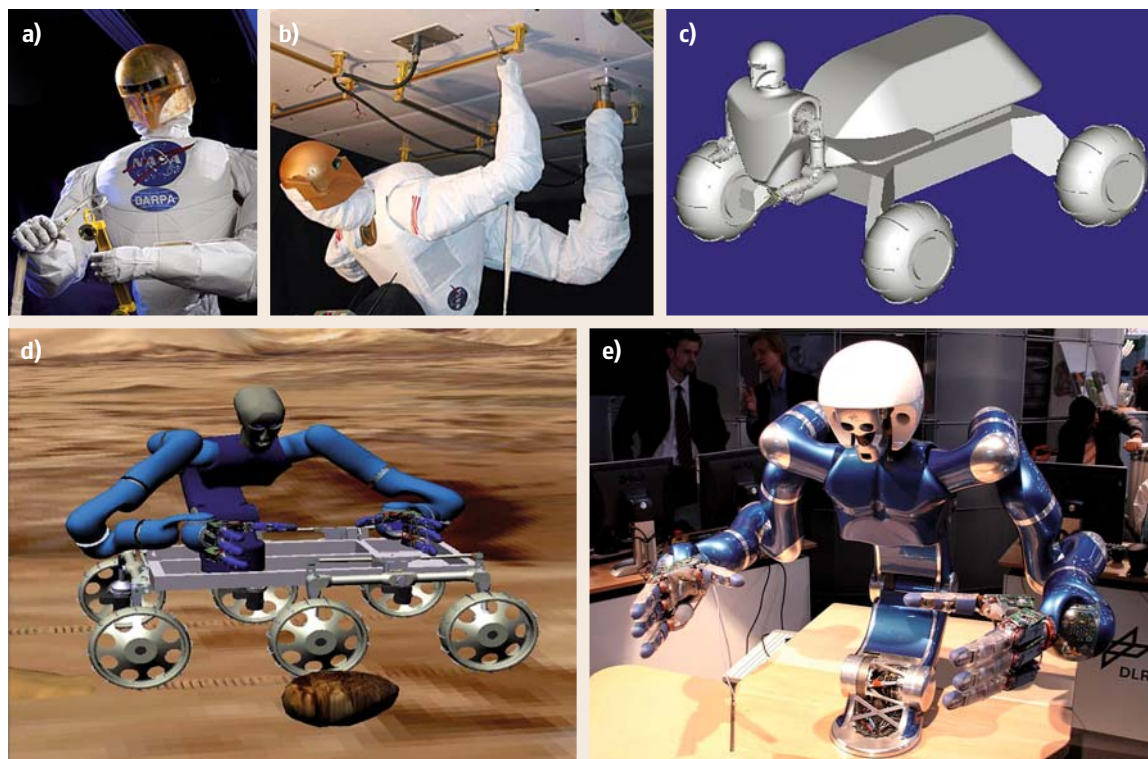


Fig. 45.32a–e Robonaut: (a,b) current model and future concepts by NASA (c), future concepts by DLR (d) which are based on the torso JUSTIN (e)

45.4.3 Aerial Platforms

There are three nonterrestrial planetary candidates for aerial robotic systems: Venus, Mars, and Titan (a moon of Saturn) [45.87, 88]. Venus has a very dense but hot atmosphere (460°C and 65 kg/m^3 at the surface) and so can easily float relatively heavy payloads. Mars has a very thin and cold atmosphere (somewhat variable but often -100°C and 0.02 kg/m^3). Titan has an atmosphere even colder than that of Mars ($\approx 100\text{ K}$) but about 50% denser than Earth's atmosphere. Thus very different vehicles have been envisioned for the three candidate mission targets. On Venus, buoyant devices are generally considered, especially those that can continuously or periodically rise high enough to reach moderate temperatures where conventional electronics can survive. One candidate is to use a phase-change fluid as part of the buoyant system, so that the fluid can condense in the cool upper atmosphere and be trapped in a pressure vessel, causing a loss of buoyancy and allowing the vehicle to descend, possibly all the way to the surface. After a brief stay and before the heat flux to the interior of

the device destroys all the sensitive equipment, a valve would be opened so that the phase-change fluid can evaporate, increasing the buoyancy and allowing the craft to ascend to the cool upper atmosphere. After a suitable period of heat rejection into this cool zone, the process can be repeated, perhaps indefinitely. The density of the Venus atmosphere is sufficiently high that powered dirigibles can be used, so that the buoyant vehicles can use propulsion and steering to reach particular locations in the atmosphere or on the surface [45.89].

In contrast, the Mars atmosphere is too thin for powered dirigibles to work (at least with the power-to-weight ratio of any current propulsion technology). Balloon aerobots could be deployed in the Mars atmosphere and could ascend and descend, but probably could not be steered precisely to specific locations, at least not by use of a propulsion system. Polar balloons could circumnavigate either pole many times or equatorial balloons could make one partial circuit around the planet, until they impact the *Tharsis Bulge*, a North-South string of high-altitude volcanoes that represents an essentially impenetrable barrier to any equatorial bal-

loon with a reasonable payload. Because of the problems with *lighter-than-air* vehicles in the thin Martian atmosphere, there has been considerable study of airplanes for use in exploring Mars. Aircraft can be designed to have reasonable lift-to-drag ratios in the Mars atmosphere, so that their performance is not too different from airplanes on Earth. Most often considered are gliders that deploy directly from an aeroshell that comes into the Mars atmosphere at hypersonic velocity and then proceed to glide hundreds or a thousand kilometers before impact. One common mission concept is to fly down the great Valles Marineris canyon, taking high-resolution imagery and spectrometry of the walls of that canyon. Powered aircraft have also been considered, including those that land and regenerate their propellant (e.g., using solar power and atmospheric CO₂) so as to be able to make multiple flights.

On Titan, like Venus, buoyant devices are generally considered most attractive (although helicopters have been proposed). Also like Venus, the atmosphere on Titan contains many obscuring particles and aerosols so that high-resolution imaging over a broad spectrum is only possible by getting close to the surface. This makes balloons or powered dirigibles very attractive. On Venus the extreme surface temperature makes it challenging to make a surface vehicle operate for an extended duration. On Titan, there is a significant risk that some sort of hydrocarbon *goo* exists on the surface that might foul any surface vehicle. Thus both Titan and Venus are considered especially attractive targets for the use of aerobots, especially in the form of powered dirigibles. Navigation of such aerobots would presumably be accomplished primarily by sensing the terrain and navigating relative to any landmarks that can be discerned. When these vehicles operate in the upper atmosphere, they can augment their position knowledge by means of sun or star tracking (as referenced to the local vertical). Deeper in the atmosphere, this may not be possible. One key issue is whether direct communications to Earth are envisioned or relay via satellite. If there is a satellite in orbit, it can provide considerable radio-navigation assistance and relatively frequent communications when the aerobot is on the side away from the Earth (both Venus and Titan spin very slowly). However, a satellite relay is expensive, so the least expensive options require that the dirigible have a large high-gain antenna (usually presumed to be inside the gas bag). Radio-based servo pointing at the Earth will provide precise navigation information (again along with precise measurements of local vertical). However, when the aerobot goes out of sight beyond the limb of the planet, it may spend days

or weeks out of communications with the Earth. This is probably the situation calling for the highest degree of autonomy of any that have been envisioned in robotic planetary exploration of the solar system.

45.4.4 Subsurface Platforms

Subsurface exploration of planetary bodies holds great promise: it is believed that a liquid-water aquifer may exist at significant depths on Mars, Europa, or Ganymede which probably represent the best possible locations within the solar system to look for extant (as opposed to extinct) extraterrestrial life. Also, in the lunar polar *dark craters* there is some evidence of the existence of water ice or other volatiles and there may exist a layered geologic record of impacts in the Earth–Moon system in these cold traps. Even access to a depth of a few meters holds the promise of reaching pristine scientific samples that have not been exposed to thermal cycling or ionizing radiation [45.90].

The prevailing wisdom has been that traditional sorts of drilling rigs are required to access these places, involving drill towers, multisegmented drill strings, large robotic systems to serve the function of a terrestrial drilling crew, and large power systems. Also, terrestrial drilling is usually done using large amounts of fluids (water, air or mud) to flush away cuttings and to cool and lubricate the cutter. The NASA Mars Technology Program has funded contractors that have demonstrated reaching 10 m of depth in a realistic setting with segmented drill strings without the use of fluids. While this is much less than needed to reach the putative liquid-water aquifers, it is much more than is reachable by previous techniques [45.91].

Other approaches have been proposed such as *moles* or *inchworms* that could be relatively self-contained and yet reach great depths without the mass and complexity of a large drill tower and segmented drill string. A key issue is that it appears that the needed energy cannot be stored onboard such self-contained drills, at least if it is stored as chemical energy. This is because drilling through terrain requires that some of the chemical bonds that hold the terrain together be broken and so if the energy of chemical bonds is used to provide that power, then a given volume of chemical energy storage can only advance some fixed ratio of its length into the terrain, where the ratio is determined by the efficiency in taking bond energies of one sort to break bonds of a different sort. Based on these considerations, it appears unlikely that a completely self-contained subsurface vehicle could advance more

than perhaps a hundred times its own length. Unless nuclear power sources are considered (and they have been), this requires some sort of tether to the surface to provide a nearly unlimited source of energy. Another problem for subsurface vehicles is that rock tends to expand when it is pulverized (in a process called *comminution*). Nonporous rock typically expands in volume by a few tens of percent when excavated, which means that fully self-contained subsurface vehicles have a severe *conservation of volume* problem. In principle the rock can be compressed back into its original volume, but this generally requires pressures much greater than the compressive strength of the original rock. The energy required to do this is much larger than the energy

required to excavate the rock in the first place and would become the dominant use of energy in an already energy-intensive effort.

As a result, it is generally assumed that any subsurface vehicle must keep some access tunnel open to the surface so that the excess volume of cuttings can be transported out. If this tunnel is available, then it seems that a means for getting power from the surface is also available, so that self-contained nuclear power is not needed. Subsurface vehicles with diameters as small as one or a few centimeters have been proposed that could potentially reach great depths within the mass and power constraints of feasible planetary robotic exploration missions.

45.5 Conclusions and Further Reading

Space robotics as a field is still in its infancy. The speed-of-light delays inherent to remote space operations makes the master–slave teleoperation approach that has been very useful in the undersea and nuclear industries problematic. Space robotics lacks the highly repetitive operations in a tightly structured environment that characterize industrial robotics. Hardware handled by space robots is very delicate and expensive. All three of these considerations have led to the fact that relatively few space robots have been flown, they have been very slow in operation, and only a small variety of tasks have been attempted. Nonetheless, the potential rewards of space robotics are great – exploring the solar system, creating vast space telescopes that may unlock the secrets of the universe, and enabling any viable space industries all seem to require major use of space robots. The scale of the solar system is not so great (a few

light-hours) that human intelligence cannot always supplement even the most remote space robot that becomes confused or stuck. Indeed, for the Moon (with only a few seconds of time delay) it seems that hazard avoidance and reliable closure of force–feedback loops is all that is required to make a highly useful robotic system. For Mars (with tens of minutes of time delay), along with hazard avoidance and force loop closure, it seems that robust anomaly detection (with modest reflexive *safing* procedures) and perhaps scientific-novelty detection are probably all that is needed. High levels of autonomy are enhancing but not enabling for work in the inner solar system and become more and more desirable for robots that are sent farther into the outer solar system.

For further reading, the following materials are suggested [45.92–97].

References

- 45.1 D. L. Akin, M. L. Minsky, E. D. Thiel, C. R. Curtzman: Space applications of automation, robotics and machine intelligence systems (ARAMIS) phase II, NASA-CR-3734-3736 (1983)
- 45.2 C.G. Wagner-Bartak, J.A. Middleton, J.A. Hunter: Shuttle remote manipulator system hardware test facility, 11th Space Simulation Conf. (1980) pp. 79–94, NASA CP-2150
- 45.3 S. Greaves, K. Boyle, N. Doshewnek: Orbiter boom sensor system and shuttle return to flight: operations analyses, AIAA Guidance Navigation Contr. Conf. Exhibit (San Francisco 2005) p. 5986
- 45.4 C. Crane, J. Duffy, T. Carnahan: A kinematic analysis of the space station remote manipulator system (SSRMS), *J. Robot. Syst.* **8**, 637–658 (1991)
- 45.5 M.F. Stieber, C.P. Trudel, D.G. Hunter: Robotic systems for the international space station, *Proc. 1997 IEEE Int. Conf. Robot. Autom.* (1997) pp. 3068–3073
- 45.6 D. Bassett, A. Abramovici: Special purpose dexterous manipulator (SPDM) requirements verification, *Proc. 5th Int. Symp. Artif. Intell. Robot. Autom. Space* (1999) pp. 43–48, ESA SP-440
- 45.7 R. Boumans, C. Heemskerk: The european robotic arm for the international space station, *Robot. Auton. Syst.* **23**(1), 17–27 (1998)

- 45.8 P. Laryssa, E. Lindsay, O. Layi, O. Marius, K. Nara, L. Aris, T. Ed: International space station robotics: a comparative study of ERA, JEMRMS and MSS, Proc. 7th ESA Workshop Adv. Space Technol. Robot. Autom. ASTRA (ESTEC, Noordwijk 2002)
- 45.9 C. Preusche, D. Reintsema, K. Landzettel, G. Hirzinger: Robotics component verification on iss rokviss – preliminary results for telepresence, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Beijing 2006) pp. 4595–4601
- 45.10 K. Landzettel, A. Albu-Schaffer, C. Preusche, D. Reintsema, B. Rebele: G. Hirzinger: Robotic on-orbit servicing – dlr's experience and perspective, Proc. of IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Beijing 2006) pp. 4587–4594
- 45.11 T. Matsueda, K. Kuraoka, K. Goma, T. Sumi, R. Okamura: JEMRMS system design and development status, Proc. IEEE National Telesyst. Conf. (1991) pp. 391–395
- 45.12 S. Doi, Y. Wakabayashi, T. Matsuda, N. Satoh: JEM remote manipulator system, J. Aeronaut. Space Sci. Japan **50**(576), 7–14 (2002)
- 45.13 H. Morimoto, N. Satoh, Y. Wakabayashi, M. Hayashi, Y. Aiko: Performance of Japanese robotic arms of the international space station, 15th IFAC World Congress (2002)
- 45.14 G. Hirzinger, B. Brunner, J. Dietrich, J. Heindl: Sensor-based space robotics–ROTEX and its telerobotic features, IEEE Trans. Robot. Autom. **9**(5), 649–663 (1993)
- 45.15 M. Oda et al.: ETS–VII, space robot in-orbit experiment satellite, Proc. 1996 IEEE Int. Conf. Robot. Autom. (1996) pp. 739–744
- 45.16 K. Yoshida: Engineering test satellite vii flight experiments for space robot dynamics and control: theories on laboratory test beds ten years ago, now in orbit, Int. J. Robot. Res. **22**(5), 321–335 (2003)
- 45.17 K. Landzettel, B. Brunner, G. Hirzinger, R. Lampariello, G. Schreiber, B.–M. Steinmetz: A unified ground control and programming methodology for space robotics applications – demonstrations on ETS–VII, Proc. Int. Symp. Robot. (Montreal 2000) pp. 422–427
- 45.18 J.C. Parrish, D.L. Akin, G.G. Gefke: The ranger telerobotic shuttle experiment: implications for operational EVA/robotic cooperation, Proc. SAE Int. Conf. Environmental Syst. (Toulouse 2000)
- 45.19 J. Shoemaker, M. Wright: Orbital express space operations architecture program. Space systems technology and operations, Proc. SPIE, Vol. 5088, ed. by P. Tchoryk Jr., J. Shoemaker, (2003) pp. 1–9
- 45.20 A.P. Vinogradov: *Lunokhod 1 Mobile Lunar Laboratory* (JPRS, Moscow 1971), JPRS identification number 54525
- 45.21 A. Mishkin: *Sojourner: An Insider's View of the Mars Pathfinder Mission* (Berkley Books, New York 2004)
- 45.22 B. Wilcox, T. Nguyen: Sojourner on mars and lessons learned for future planetary rovers, 28th Int. Conf. Env. Syst. (Danvers 1998)
- 45.23 M. Maimone, A. Johnson, Y. Cheng, R. Willson, L. Matthies: Autonomous navigation results from the Mars exploration rover (MER) mission, Proc. 9th Int. Symp. Experimental Robot. (ISER) (Singapore 2004)
- 45.24 M. Maimone, Y. Cheng, L. Matthies: Two years of visual odometry on the Mars exploration rovers, J. Field Robot. **24**(3), 169–186 (2007)
- 45.25 M.G. Bekker: *Off-The-Road Locomotion* (Univ. Michigan Press, East Lausing 1960)
- 45.26 M.G. Bekker: *Introduction to Terrain-Vehicle Systems* (Univ Michigan Press, East Lausing 1960)
- 45.27 R.A. Lewis, A.K. Bejczy: Planning considerations for a roving robot with arm, IJCAI (1973) pp. 308–316
- 45.28 A. Thompson: The navigation system of the JPL robot, IJCAI (1977) pp. 749–757
- 45.29 E. J. M. West, W. Trautwein: Operational Loopwheel Suspension Systems for Mars Rover Demonstration Model Loopwheel failure report. Document LMSC-HREC TR D568859, (Huntsville Research & Engineering Center, Huntsville 1979) Available at http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19790011992_1979011992.pdf
- 45.30 Y. Yakimovsky, R.T. Cunningham: A system for extracting three-dimensional measurements from a stereo pair of TV cameras, Comp. Graph. Image Proc. **7**, 195–210 (1978)
- 45.31 E. Krotkov, J. Bares, T. Kanade, T. Mitchell, R. Simmons, W. Whittaker: Ambler: a six-legged planetary rover, 5th Int. Conf. Adv. Robot. 1991 Robots Unstructured Env., Vol. 1 (1991) pp. 717–722
- 45.32 J. Bares, W. Whittaker: Walking robot with a circulating gait, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. Towards New Frontier Appl. (IROS '90), Vol. 2 (1990) pp. 809–816
- 45.33 B. Wilcox, D. Gennery: A Mars rover for the 1990s, J. Br. Interplanet. Soc. **40**, 484–488 (1987)
- 45.34 B. Wilcox, L. Matthies, D. Gennery: Robotic vehicles for planetary exploration, Proc. Int. Conf. Robot. Autom. (Nice 1992)
- 45.35 D. Gennery, T. Litwin, B. Wilcox, B. Bon: Sensing and perception research for space telerobotics at JPL, IEEE Int. Conf. Robot. Autom. (Raleigh 1987) pp. 311–317
- 45.36 J. Balaram, S. Hayati: A supervisory telerobotics testbed for unstructured environments, J. Robot. Syst. **9**(2), 261–280 (1992)
- 45.37 G. Giral, L. Boissier: The French planetary rover VAP: concept and current developments, IEEE Int. Conf. Intell. Robot. Syst. (IROS'92) (Raleigh 1992) pp. 1391–1398, LAAS Report No. 92227
- 45.38 R. Chatila, S. Lacroiz, G. Giral: A case study in machine intelligence: adaptive autonomous space rovers, Int. Conf. Field Service Robot. (FSR'97) (Cannberra 1997) pp. 101–108, LAAS Report No. 97463

- 45.39 A. Castano, A. Fukunaga, J. Biesiadecki, L. Neakrase, P. Whelley, R. Greeley, M. Lemmon, R. Castano, S. Chien: Autonomous detection of dust devils and clouds on Mars, Proc. IEEE Int. Conf. Image Proc. (Atlanta 2006) pp. 2765–2768
- 45.40 L. Matthies: Stereo vision for planetary rovers: stochastic modeling to near real-time implementation, Int. J. Comp. Vision **8**(1), 71–91 (1992)
- 45.41 K. Yoshida, D.N. Nenchev, M. Uchiyama: Moving base robotics and reaction management control, Robot. Res.: 7th Int. Symp., ed. by G. Giralt, G. Hirzinger (Springer, New York 1996) pp. 101–109
- 45.42 J. Russakow, S.M. Rock, O. Khatib: An operational space formulation for a free-flying multi-arm space robot, 4th Int. Symp. Exp. Robot. (1995) pp. 448–457
- 45.43 Y. Umetani, K. Yoshida: Continuous path control of space manipulators mounted on OMV, Acta Astro. **15**(12), 981–986 (1987), presented at the 37th IAF Conf, Oct. 1986
- 45.44 Y. Umetani, K. Yoshida: Resolved motion rate control of space manipulators with generalized jacobian matrix, IEEE Trans. Robot. Autom. **5**(3), 303–314 (1989)
- 45.45 Y. Xu, T. Kanade (eds.): *Space Robotics: Dynamics and Control* (Kluwer Academic, Boston 1993)
- 45.46 K. Yoshida: Impact dynamics representation and control with extended inversed inertia tensor for space manipulators, Robot. Res. 6th Int. Symp., ed. by T. Kanade, R. Paul (1994) pp. 453–463
- 45.47 K. Yoshida: Experimental study on the dynamics and control of a space robot with the experimental free-floating robot satellite (EFFORTS) simulators, Adv. Robot. **9**(6), 583–602 (1995)
- 45.48 Y. Nakamura, R. Mukherjee: Nonholonomic path planning of space robots via a bidirectional approach, IEEE Trans. Robot. Autom. **7**(4), 500–514 (1991)
- 45.49 S. Dubowsky, M. Torres: Minimizing attitude control fuel in space manipulator systems, Proc. Int. Symp. AI Robot. Autom. (i-SAIRAS) (1990) pp. 259–262
- 45.50 S. Dubowsky, M. Torres: Path planning for space manipulators to minimize spacecraft attitude disturbances, Proc. IEEE Int. Conf. Robot. Autom., Vol. 3 (Sacramento 1991) pp. 2522–2528
- 45.51 K. Yoshida: Practical coordination control between satellite attitude and manipulator reaction dynamics based on computed momentum concept, Proc. 1994 IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Munich 1994) pp. 1578–1585
- 45.52 M. Oda: Coordinated control of spacecraft attitude and its manipulator, Proc. 1996 IEEE Int. Conf. Robot. Autom. (1996) pp. 732–738
- 45.53 Z. Vafa, S. Dubowsky: On the dynamics of manipulators in space using the virtual manipulator approach, Proc. IEEE Int. Conf. Robot. Autom. (1987) pp. 579–585
- 45.54 E. Papadopoulos, S. Dubowsky: Dynamic singularities in the control of free-floating space manipulators, ASME J. Dyn. Syst. Meas. Contr. **115**(1), 44–52 (1993)
- 45.55 Y. Umetani, K. Yoshida: Workspace and manipulability analysis of space manipulator, Trans. Soc. Instrum. Contr. Eng. **E-1**(1), 116–123 (2001)
- 45.56 D.N. Nenchev, Y. Umetani, K. Yoshida: Analysis of a redundant free-flying spacecraft/manipulator system, IEEE Trans. Robot. Autom. **8**(1), 1–6 (1992)
- 45.57 D.N. Nenchev, K. Yoshida, P. Vichitkulsawat, M. Uchiyama: Reaction null-space control of flexible structure mounted manipulator systems, IEEE Trans. Robot. Autom. **15**(6), 1011–1023 (1999)
- 45.58 D.N. Nenchev, K. Yoshida, M. Uchiyama: Reaction null-space based control of flexible structure mounted manipulator systems, Proc. IEEE 35th CDC (1996) pp. 4118–4123
- 45.59 W.J. Book, S.H. Lee: Vibration control of a large flexible manipulator by a small robotic arm, Proc. Am. Contr. Conf. (Pittsburgh 1989)
- 45.60 M.A. Torres: Modelling, path-planning and control of space manipulators: the coupling map concept. Ph.D. Thesis (MIT, Cambridge 1993)
- 45.61 S. Abiko, K. Yoshida: An effective control strategy of japanese experimental module remote manipulator system (JEMRMS) using coupled and un-coupled dynamics, Proc. 7th Int. Symp. Artif. Intell. Robot. Autom. Space Paper AS18 (CD-ROM) (Nara 2003)
- 45.62 S. Abiko: Dynamics and Control for a Macro-Micro Manipulator System Mounted on the International Space Station. Ph.D. Thesis (Tohoku University, Sendai 2005)
- 45.63 S. Abiko, K. Yoshida: On-line parameter identification of a payload handled by flexible based manipulator, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'04) (Sendai 2004) pp. 2930–2935
- 45.64 S. Abiko, K. Yoshida: An adaptive control of a space manipulator for vibration suppression, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'05) (Edmonton 2005)
- 45.65 G. Gilardi, I. Shraf: Literature survey of contact dynamics modeling, Mech. Machine Theory **37**, 1213–1239 (2002)
- 45.66 K. Yoshida, H. Nakanishi, H. Ueno, N. Inaba, T. Nishimaki, M. Oda: Dynamics, control, and impedance matching for robotic capture of a non-cooperative satellite, Adv. Robot. **18**(2), 175–198 (2004)
- 45.67 H. Nakanishi, K. Yoshida: Impedance control of free-flying space robot for orbital servicing, J. Robot. Mechatron. **18**(5), 608–617 (2006)
- 45.68 P.M. Pathak, A. Mukherjee, A. DasGupta: Impedance control of space robots using passive degrees of freedom in controller domain, J. Dyn. Syst. Meas. Contr. **127**, 564–578 (2006)

- 45.69 S. Abiko, R. Lampariello, G. Hirzinger: Impedance control for a free-floating robot in the grasping of a tumbling target with parameter uncertainty, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS '06) (Beijing 2006)
- 45.70 K. Iagnemma, H. Shibly, S. Dubowsky: On-Line traction parameter estimation for planetary rovers, Proc. IEEE Int. Conf. Robot. Autom. (ICRA '02) (Washington 2002)
- 45.71 K. Yoshida, H. Hamano: Motion dynamics of a rover with slip-based traction model, Proc. IEEE Int. Conf. Robot. Autom. (ICRA '02) (Washington 2002)
- 45.72 A. Jain, J. Guineau, C. Lim, W. Lincoln, M. Pomerantz, G. Sohl, R. Steele: Roams: planetary surface rover simulation environment, Proc. 7th Int. Symp. Artif. Intell. Robot. Autom. Space (ISAIRAS '03) (Nara 2003)
- 45.73 K. Yoshida, G. Ishigami: Steering characteristics of a rigid wheel for exploration on loose soil, Proc. IEEE Int. Conf. Intell. Robot. Syst. (IROS '04) (Sendai 2004)
- 45.74 R. Bauer, W. Leung, T. Barfoot: Experimental and simulation results of wheel-soil interaction for planetary rovers, Proc. IEEE Int. Conf. Intell. Robot. Syst. (IROS '05) (Edmonton 2005)
- 45.75 G. Ishigami, K. Yoshida: Steering characteristics of an exploration rover on loose soil based on all-wheel dynamics model, Proc. IEEE Int. Conf. Intell. Robot. Syst. (IROS '05) (Edmonton 2005)
- 45.76 A. Ellery, N. Patel, R. Bertrand, J. Dalcomo: Exomars rover chassis analysis and design, Proc. 8th Int. Symp. Artif. Intell. Robot. Autom. Space (ISAIRAS '05) (Munich 2005)
- 45.77 A. Gibbesch, B. Schäfer: Multibody system modelling and simulation of planetary rover mobility on soft terrain, Proc. 8th Int. Symp. Artif. Intell. Robot. Autom. Space (ISAIRAS '05) (Munich 2005)
- 45.78 G. Ishigami, A. Miwa, K. Nagatani, K. Yoshida: Terramechanics-based model for steering maneuver of planetary exploration rovers on loose soil, J. Field Robot. **24**(3), 233–250 (2007)
- 45.79 J.Y. Wong: *Theory of Ground Vehicles* (Wiley, New York 1978)
- 45.80 K. Iagnemma, S. Dubowsky: *Mobile Robots in Rough Terrain: Estimation, Motion Planning, and Control With Application to Planetary Rovers*, Springer Tracts in Advanced Robotics, Vol. 12 (Springer, Berlin, Heidelberg 2004)
- 45.81 K. Yoshida, T. Watanabe, N. Mizuno, G. Ishigami: Terramechanics-based analysis and traction control of a lunar/planetary rover, Proc. Int. Conf. Field Service Robot. (FSR '03) (Yamanashi 2003)
- 45.82 J.Y. Wong, A.R. Reece: Prediction of rigid wheel performance based on the analysis of soil-wheel stresses. Part I, performance of driven rigid wheels, J. Terramechan. **4**, 81–98 (1967)
- 45.83 Z. Janosi, B. Hanamoto: The analytical determination of drawbar pull as a function of slip for tracked vehicle in deformable soils, Proc 1st Int. Conf. Terrain-Vehicle Syst. (Torino 1961)
- 45.84 E. Hegedus: A simplified method for the determination of bulldozing resistance, Land Locomotion Research Laboratory, Army Tank Automotive Command Report, Vol. 61, 1960
- 45.85 I. Rekleitis, E. Martin, G. Rouleau, R. L'Archeveque, K. Parsa, E. Dupuis: Autonomous capture of a tumbling satellite, J. Field Robot. **24**(4), 275–296 (2007)
- 45.86 R. Ambrose: <http://robonaut.jsc.nasa.gov/> Jan (2008)
- 45.87 J. Blamont: Balloons on other planets, Adv. Space Res. **1**, 63–69 (1981)
- 45.88 J.A. Cutts, K.T. Nock, J.A. Jones, G. Rodriguez, J. Balaram, G.E. Powell, S.P. Synott: Aerovehicles for planetary exploration, IEEE Int. Conf. Robot. Autom. (Nagoya 1995)
- 45.89 M.K. Heun, J.A. Jones, J.L. Hall: Gondola design for Venus deep-atmosphere aerobot operations, AIAA, 36th Aerospace Sci. Meeting Exhibit, 1998 ASME Wind Energy Symp. (Reno 1998)
- 45.90 S. Miller, J. Essmiller, D. Beaty: Mars deep drill – a mission concept for the next decade, Space Conf. Exhibit (San Diego 2004) pp. 2004–6048, AIAA
- 45.91 S. Mukherjee, P. Bartlett, B. Glass, J. Guerrero, S. Stanley: Technologies for exploring the martian subsurface, IEEE Aerospace Conf. Proc. (Big Sky 2006), No. 1349
- 45.92 S.B. Skaar: Teleoperation and robotics in space. In: *Progress in Astronautics and Aeronautics Series*, ed. by C.F. Ruoff (AIAA Technology and Industrial Arts, Notre Dame 1994), ISBN 1563470950
- 45.93 G. F. Bekey: Assessment of international research and development in robotics a study report of the national science foundation/NASA study, available at <http://www.wtec.org/robotics/> (soon to be published in book form from World Scientific Publishing in Singapore).
- 45.94 Assessment of Options for Extending the Life of the Hubble Space Telescope, National Research Council: Final Report, (National Academies Press, 2005) ISBN-10: 0-309-09530-1, ISBN-13: 978-0-309-09530-3
- 45.95 A.M. Howard, E.W. Tunstel: *Intelligence for Space Robotics* (TSI Press, San Antonio 2006), ISBN 1-889335-26-6/1-889335-29-0.
- 45.96 T.B. Sheridan: Space teleoperation through time delay: review and prognosis, IEEE Trans. Robot. Autom. **9**(5), 592–606 (1993)
- 45.97 J. Field Robot. Special Issue on Space Robotics, Part I–Part III, **24**(3–5), 167–434 (2007)

46. Robotics in Agriculture and Forestry

John Billingsley, Arto Visala, Mark Dunn

In agriculture and forestry, robotics has made a substantial impact. Farmers are conscious of their need for automatic vehicle guidance to minimize damage to the growing zone of their soil. Automatic sensing, handling, and processing of produce are now commonplace, while there is substantial instrumentation and mechanization of livestock procedures. In forestry, legged harvesters have not yet seen great success in their application, but the automation of trimming and forwarding with simultaneous localization and mapping techniques will change the industry in the future.

Some impressive developments in walking forestry harvesters are presented, including machines targeted towards the difficult terrain of the Scandinavian forests. More-conventional cut-to-length harvesters are also highly automated, while operations such as *delimbing* must be carried out at speed. Before complete autonomous harvesting becomes possible, some of the localization and mapping techniques that are described must come to fruition.

The combination of machine vision with global positioning by satellite (GPS) allows a tractor to follow a row of crops, performing a headland turn at the end of the row. The history of a series of projects is outlined, leading to the present outcome that is in the process of being commercialized. Another project that is based on machine vision relates to the location of macadamia nuts. To select which trees should be propagated, it is necessary to attribute fallen nuts to the correct tree. Color sorting and grading of produce is not a matter of sensing alone, but involves a measure of produce handling that puts it at the fringe of robotics.

Automated milking parlours have proved their worth. However success has eluded some other projects described here, such as automated sheep-shearing and an automated abattoir. Another project is presented that literally sorts the sheep

46.1 Definitions	1066
46.1.1 Technological Developments	1066
46.2 Forestry	1066
46.2.1 Overview	1066
46.2.2 Robot Locomotion in Forestry	1067
46.2.3 Forestry Automation	1068
46.2.4 Machine Perception and SLAM in the Forest Environment	1068
46.2.5 Autonomy, Teleoperation, and Fleet Management	1069
46.2.6 Autonomous Robots for Silviculture and Treatment	1069
46.2.7 Forestry Conclusions	1070
46.3 Broad Acre Applications	1070
46.3.1 An Overview of Automatic Guidance	1070
46.3.2 Sowing, Weeding, Spraying, and Broad-Acre Harvesting	1071
46.4 Horticulture	1071
46.4.1 Picking of Fruit and Vegetables	1071
46.4.2 Color Sorting and Produce Grading	1072
46.5 Livestock	1072
46.5.1 Robot Milking	1072
46.5.2 Sheep Shearing	1073
46.5.3 Slaughtering	1073
46.5.4 Livestock Inspection	1074
46.5.5 Robotic Animals	1074
46.6 Unmanned Vehicles	1075
46.7 Conclusions and Future Directions	1075
References	1075

from the goats, using a swinging gate to separate different species using machine vision so that feral species are excluded from watering holes in the dry Australian outback.

Although robotics is making rapid inroads into these areas, they are still a fruitful source of application projects, some sufficiently demanding to require the development of new theoretical techniques.

46.1 Definitions

The boundaries of agriculture are not clear cut. Preparation of the soil, planting, cultivating, watering, spraying, and harvesting are evidently included, but how far into the post-harvest processes of trimming, sorting, and grading can we go before they merge with food preparation? Similarly, animal-based activities can extend beyond milking and shearing to slaughtering and butchering, at the start of a long chain of operations leading to the appearance of processed food or manufactured garments on the supermarket shelves.

Robotics made its first real appearance in the manufacturing industry with the adoption of the name *Robot* for the serial manipulator. Here manipulation and its related kinematics formed the core of the art, later developing into intelligent automation. When the essentials of robotics are applied to the much more significant industry of agriculture, however, the emphasis must be placed much more heavily on sensing than on manipulation. When a tractor is steered automatically or a gate is closed because a feral pig has been recognized, the answer to the question *Is this robotics?* is not cut and dried. Without including such applications, however, many of the advances in agricultural automation would be overlooked.

Forestry deals with the harvesting of wood. Forestry machines are today still mostly directly controlled by human drivers, with the help of distributed controller area network (CAN)-based automation systems, but these machines will become more autonomous and robot-like in the future. The machines will have a perception system, which will map the trees and localize the machine. Information about the forest stand can thus be collected so that operation of the semiautomatic crane and loader becomes possible, together with steering and driving. Most of the forests in the Northern hemisphere are natural forests that are cared for. Particularly in Nordic countries, there is efficient and sustainable silviculture, the science, art, and practice of caring for forests for human exploitation. Stands of forest trees are thinned before clear cutting and replacement trees are either seeded

or planted. Autonomous machines for silviculture will be an important research area in the future.

46.1.1 Technological Developments

Perhaps the greatest impact on agriculture has been through the farmer's growing awareness of computer power. Mobile computing can log the yield during harvesting, relating it to a precision map of the property. Tasks that had been merely mechanized can now be synchronized and automated, such as the lifting of spray booms and implements when turning.

Global positioning by satellite (GPS) has been seized on for mapping and guidance operations.

Generic radio communication techniques use protocols that might be common to mobile telephone technology or networked systems such as *Zigbee* [46.1]. These systems allow remote monitoring of gates, livestock, or equipment and automatic replenishment of water troughs. They can equally contribute to fleet management and information transfer in a forestry situation. Other radio technology in the form of transponders can see each animal tagged and identified to support activities such as milking and tracking *from the paddock to the plate*.

Much farming machinery has long been hydraulically powered, but the addition of digitally controlled valves opens the way for automated steering and other robot operations.

With computing power comes the ability to analyze images from cameras, which are becoming ever cheaper. Vision sensing has pervaded sorting operations, but now it extends to vision guidance and the recognition of animals to permit or deny them access to watering points.

It is unlikely that large tractors will ever be allowed to roam unmanned because of the risk of legal repercussions, but the day when cooperating teams of small autonomous *farmhand robots* will be seen in the fields is drawing ever closer [46.2].

46.2 Forestry

46.2.1 Overview

Forestry has progressed from manual to machine harvesting, such that by the end of the 1990s some 95% of

the wood in Nordic countries was harvested with machines. Now the challenge of introducing robotics is being addressed. A typical harvester machine is shown in Fig. 46.1 and a forwarder, the machine that transports



Fig. 46.1 Cut-to-length (CTL) harvester made by Ponsse Oy Ltd. [46.3]

wood from the felling site to the roadside, is shown in Fig. 46.2.

Nowadays all new Nordic harvesters are controlled with a CAN-based distributed control system and information system, with GPS localization utilizing mobile communication networks to transfer data relating to harvesting. This forms a basic platform for adding more enhanced control functions. However, in a forest en-



Fig. 46.2 CTL forwarder by Ponsse Oy Ltd. [46.3]

vironment, GPS does not work well enough for the exact localization needed in machine control; simultaneous localization and mapping (SLAM) algorithms are needed.

A digital map of the forest stand and the target log assortments are downloaded to the harvester when the work begins. The diameter and volume of every log is measured when the log is *delimbed* and *bucked* (cross-cut) to the selected length. Information about the accumulated log situation is uploaded to a higher information system to manage the transportation of the wood from the side of the forest road to the mills. This logistics system incorporates trucks with digital maps and positioning systems, and is highly optimized in Nordic countries. The coordinates of every stack of different types of log are included in the information system and are delivered to the truck responsible for their collection.

46.2.2 Robot Locomotion in Forestry

Most current forestry machines use wheels as their locomotion mechanism. With active control, the machines can move in very difficult forest terrain if the slopes are not too steep. Much research and development has been performed in this area in order to stabilize and smooth locomotion with wheels over uneven terrain.

For locomotion in mountainous areas, new technological solutions such as walking are required. There are many areas in which mountainous terrain form a large



Fig. 46.3 Walking forest harvester prototype by Plustech Ltd., today part of John Deere

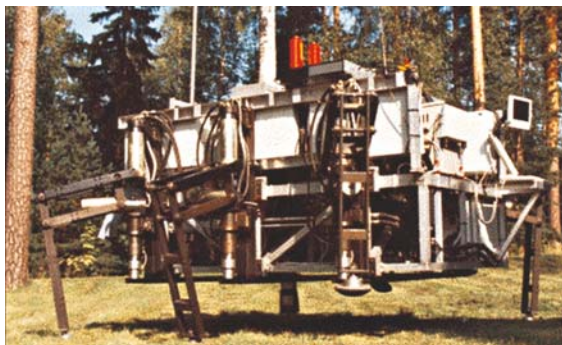


Fig. 46.4 MECANT walking machine by Halme et al. at TKK

proportion of the forest. Walking is perhaps the only safe form of locomotion on hillsides and mountainsides and has been an exciting research area in forestry robotic since the late 1980s.

In 1995, Plustech Oy published a prototype of a walking forest harvester, shown in Fig. 46.3. Plustech was owned by Timberjack Oy, which is today part of John Deere. Plustech Oy developed their walking machine independently, but most likely profited from the experience and results from a research project in which they had participated, MECANT 1989–1995. MECANT was developed at the TKK Automation Technology Laboratory led by Professor Aarne Halme [46.3]. MECANT, shown in Fig. 46.4, is a six-legged hydraulic remotely operated walking platform for studying locomotion in natural environments for work machines. Halme and Vainio at TKK published an article about forest robotics in 1998 [46.4], in which MECANT is also introduced. Newer research on locomotion at TKK has concentrated on combined walking with legs and wheels in each leg, rolking, developed in the Workpartner platform by Halme et al. [46.5].

In Japan, most of the forests are in the mountains and many research projects have been conducted on walking technology for forestry, for example at Tokyo University by Toshio Nitami and co-workers.

Operating walking harvesters on mountainsides could be risky for human operators sitting in the cabin of the machine. Such machines should be remotely controlled. Teleoperation is dealt with in more detail below.

46.2.3 Forestry Automation

Almost all cranes used in cut-to-length harvesters and grab loaders in the forwarders are today driven with

hydraulics. The control system of the crane or grab loader is based on CAN controllers with electrohydraulic valves. However, in spite of these technological possibilities, human drivers still control all movements. Some combined movements exist so that linear movements can be easily controlled, but commercial harvesters or forwarders do not have any automatic or even semiautomatic work cycles in their cranes and grab loaders.

Research to realize automatic or semiautomatic crane control has been underway since the 1980s. For example laser-pointer-assisted motion control of a forestry crane was implemented and tested in 1984 by Manninen and Halme [46.6]. The principles of interactive robotics, which could be used in connection with forestry cranes, were presented by Halme et al. [46.7]. The crane can easily be instrumented, since it resembles an industrial robot. Other enhancements can control swinging of the load.

During recent years, research has been performed to develop better control algorithms for motion control in the harvester head. The stem should be moved as fast as possible during delimbing, but no slip is allowed. The stem should be stopped at just the right place for cross cutting because the allowed tolerance in the length of the logs is only some centimeters. Koivo and Viljamaa et al. have developed well-behaved solutions for this problem [46.8].

46.2.4 Machine Perception and SLAM in the Forest Environment

In mobile robotics, simultaneous localization and mapping (SLAM) algorithms have been extensively studied. In forestry, the main task is to detect and parameterize the valuable trees among other plants and nonvaluable trees. This is a very difficult task due to illumination changes in general, bushes and young trees, and the branches of valuable trees. The perception must also work in winter-time, when there may be more than 1 m of snow that can cover the trees, and expensive forestry machines must be driven in almost complete darkness in the early morning and late evening.

With experience in using two-dimensional (2-D) laser scanners for measuring logs, work on modeling of the trees and forest on the basis of laser scanners has been performed since the 1990s. Figure 46.5 shows forest modeling results by Wernersson and Höglström [46.9]. Similar work with a three-dimensional (3-D) scanner has also been undertaken at TKK by Halme and Forsman, as shown in Fig. 46.6 [46.10]. Modeling of the forest

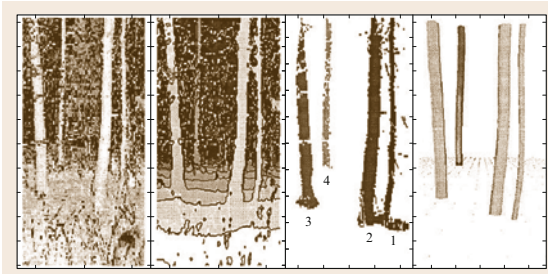


Fig. 46.5 Typical laser measurements of a typical outdoor forest scene, segmentation, and shape estimation by Wernersson [46.10]

on the basis of airborne 3-D laser scanners has been an important theme in remote sensing. However, because of the tops and branches of the trees, sufficient detail for forestry robotics cannot be measured from the air or space; this information must be supplemented with measurements in the forest terrain.

Autonomous SLAM capabilities are widely held to be one of the key features of outdoor mobile robots, including forestry robotics. Robot navigation has been an ongoing research topic for several years.

Navigation in outdoor environments is an open problem. The absence of simple features leads to the need for more-complex perception and modeling. 2-D laser range finders have become one of the most attractive sensors for localization and map building due to their accuracy and low cost. Most common laser scanners provide range and bearing information with sub-degree

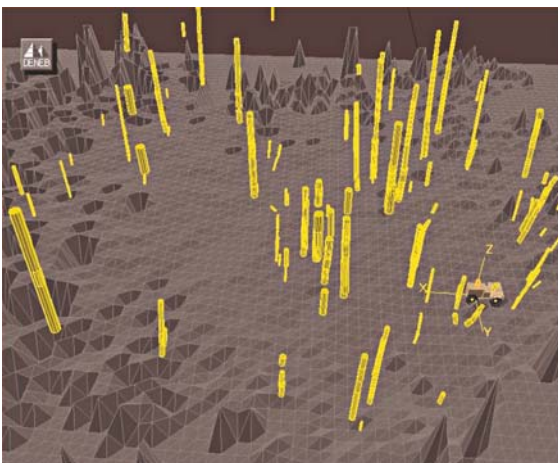


Fig. 46.6 A bird's-eye view into the combined 3-D elevation/feature model of a small forest area by Forsman and Halme [46.11]

resolution and accuracies on the order of 1–10 cm for 10–50 m ranges. Valuable work has been done, particularly in the University of Sydney by *Nebot* and *Bailey* et al. [46.11–14]. Web sources concerning forestry measurements by Australian National University are also useful [46.15].

The prospect of a fully autonomous forest harvester seems to lie far in the future. However, it is possible to advance step by step. A SLAM-based support system for the driver of the harvester seems to be reasonable and the benefits rewarding. In the ongoing Forestrix project by Visala et al. at Helsinki University of Technology TKK, forest and tree trunk measurement technologies, signal processing methods, and algorithms are studied in order to develop this kind of support system for the driver of a harvester. Semiautomatic control of forest harvesters for easy conditions is also being studied. In every forest stand, there are areas where only the human operator can control the machine. Advances in laser range finders and machine vision systems provide opportunities for new kinds of forest measurements.

46.2.5 Autonomy, Teleoperation, and Fleet Management

At Umeå University, a group led by Thomas *Helström* is developing autonomous driving of the *forwarder*. A human driver teaches the system by driving the route himself first [46.16]. This is a *real-world* study and results can be drawn from studies in the military sector.

Teleoperated loaders are already commonly used in mining. The main reasons for teleoperation are safety and efficiency, since one operator can operate several machines without being physically present in any of them. In mountainous forests, safety is a very good argument for the use of teleoperated machines. In relatively flat areas, as in Nordic countries, teleoperation can be advocated on the basis of increased efficiency. In Sweden a remote-controlled harvester operated by the driver of the forwarder has been developed by *Löfgren* et al. [46.17]. The trees are harvested directly to the cargo space of the forwarder. One remote-controlled harvester serves two forwarders.

True teleoperation over long distances requires efficient wireless communication, which can be a problem. For example live video links require quite high bandwidth, which cannot be implemented on the basis of mobile communication services existing today in the deep countryside. Teleoperated machines should there-

fore be intelligent enough that less efficient wireless communication is sufficient.

46.2.6 Autonomous Robots for Silviculture and Treatment

In Nordic countries, after the final clear-cutting of trees the ground of the forest stand is usually prepared for planting of seedlings or seeds. Seeding can be combined with automatic tilling. Planting has been partly mechanized over the last few years, using harvesters in which implanting units are installed in the tip of cranes. Planting could be robotized so that the planting plan is realized with a group of small robots employing real-time kinematics (RTK) GPS. However, there is much expertise and knowledge related to implanting it-

self, soil, and the planting site, which is difficult but not impossible to automate.

In order to automate weeding, more research is needed in this sector of forestry robotics. Pioneering research has been done in Canada by Petawawa at the National Forestry Institute [46.18].

46.2.7 Forestry Conclusions

Forestry is a demanding area for robotics. It is a harsh environment for all types of instrumentation. Reliable perception and measurement of essential objects and state parameters in real time is the bottleneck to developing more enhanced autonomous or teleoperated functions and operations in forestry machines.

46.3 Broad Acre Applications

46.3.1 An Overview of Automatic Guidance

Since the earliest horse-ploughing contests, farmers have aspired to straight lines. In Australia and other countries, this has been strengthened by the concept of *controlled traffic*. The belief is that if vehicles can be made to run in the same wheel ruts from year to year, the least damage will be done to the growing zone of the soil. These factors lend strength to the desirability of an automatic guidance system that has an accuracy of a few centimeters.

Methods that were considered included buried cables, but by the early 1990s it was clear that machine vision and the anticipated GPS offered the greatest promise. Workers at the University of Illinois had already researched means of deriving guidance information from row-crop images, but in those days computing power was limited and image capture interfaces expensive.

With a simple frame-grabber that captured a sparse binary image, researchers in Southern Queensland achieved automatic steering to centimeter accuracy. In a mere five years, low-cost video capture systems became available which enabled a colour-based system to be developed, tested, and brought to market [46.19]. The central algorithm was based on a regression fit of lines to plants seen in *keyholes* that moved to track the rows. Discrimination of *plant/not plant* employed a variable level that tracked a farmer-entered estimate of percentage ground cover. Steering was implemented by a submodule with an embedded microprocessor that switched

valves in the hydraulic steering system, the loop being closed by a Hall-effect steering sensor. Figure 46.7 shows the row-fitting algorithm in action.

Despite great field success, the marketing effort through a major US tractor-maker saw few units sold. It is thought that the price was set too low, so that dealers were reluctant to take responsibility for high-technology equipment in sites that were mainly remote. Another handicap was the growing publicity for GPS guidance.

In those days, to achieve an accuracy that was claimed to be better than 10 cm GPS systems needed to be based on two-band receivers with a ground



Fig. 46.7 Camera view with rows identified

base-station and [RTK](#). Nobody could accuse the market-leading systems of being underpriced, with price tags of as much as 80 000 Australian dollars. However the initial products did not provide automatic guidance but merely displayed a guide bar for a human driver. The steering submodule that had been developed for vision guidance was sold in quantity to adapt the [GPS](#) systems for automatic operation. These *steering ready kits* sold for double the end-user price of the entire vision guidance system of which they had been a minor part.

With the convergence of computing and entertainment, cameras can now be directly interfaced through universal serial bus (USB) ports. Processing power and software are abundant. Differential carrier-based techniques allow low-cost [GPS](#) receivers to offer centimeter displacement tracking and the new generation of systems combine vision, [GPS](#), and inertial sensors.

Whether tractors will ever be fully autonomous lies more in the hands of the litigation lawyers than in those of the engineers.

46.3.2 Sowing, Weeding, Spraying, and Broad-Acre Harvesting

When the individual operations are considered, the technologies vary in their importance. For spraying, 0.5 m accuracy is usually sufficient. Here, however, demands of speed are at their most important, so there is a trade-off between a [GPS](#) with a once-per-second update with inertial assistance and the 5 Hz [GPS](#) that is becoming more common. For yield monitoring, where the har-

vested yield might be apportioned into 5 m squares, [GPS](#) with 0.5 m precision is also sufficient.

When *listing up*, there is little or no visual reference in the field. Precision [GPS](#) and inertial sensors have a clear lead. However, when a good furrow has been formed, planting can employ a simple mechanical *furrow follower*, a ball or wheel trailed from an arm projecting in front of the tractor.

When seedlings have emerged, machine vision offers great advantages over [GPS](#) guidance. While cultivating, any dynamic positional errors at the planting stage will be added to steering positional errors; the blades must be set further from the row if the risk of destroying plants is to be contained. Vision can track the wander in the planted rows so that only one level of errors will be involved.

When a cotton crop is ready for harvesting, simple methods can again be used. Mechanical *stem feelers* are ideal for guiding the harvester precisely along the rows and these have been successfully field-tested by the National Center for Engineering in Agriculture (NCEA). For wheat, however, vision could still be supreme. The Illinois researchers have investigated visual ways of detecting the boundary of the previous cut.

A simpler robot already in widespread use is the center-pivot irrigation device [46.20]. These systems are self-propelled, irrigating an area of up to 600 acres per pass. Add-ons such as [GPS](#), moisture monitors, and even imaging devices add sensory input for decision making. In this way, water and fertilizer may be applied to specific areas of the field at specific rates dependent on conditions [46.21, 22].

46.4 Horticulture

46.4.1 Picking of Fruit and Vegetables

Once we leave broad-acre crops, harvesting can require selection and sensing. Brute force *tree shakers* might be used for picking some citrus fruit, but hand-picking is still common. Intelligent picking has presented a challenge to many robotics researchers [46.23–27].

Picking can sometimes take the form of a location or localization task, deriving a target position for the picking actuator. At other times there is an additional requirement to determine which of the fruit are ready for picking and which must be left to ripen. At present, the *automation* consists of no more than conveyor belts extending each side of a tractor while hand-pickers walk

the field, choosing which to cut, be they broccoli, rock melons, or cauliflowers.

The gathering of macadamia nuts is performed by a manually steered vehicle with a *bristle roller* which gathers up the nuts from the ground. What brings it to the attention of robotics is a localization task associated with selecting for varietal strains. It is necessary to attribute each kernel to the correct tree, meaning that the absolute position must be measured.

Cameras inspect the rollers just before the nuts are stripped, as shown in Fig. 46.8, so that the pick-up location is known precisely with respect to the vehicle, but that leaves the task of locating the vehicle. [GPS](#) is unreliable under the tree canopy, so the system



Fig. 46.8 Macadamia harvester with cameras

combines odometry with tree-trunk location using both sideways-looking visual streaming and radio frequency identification (RFID) tagging [46.28].

46.4.2 Color Sorting and Produce Grading

Some tomato growers resort to mass-harvesting tomatoes that are still green and rock hard, relying on ripening them in a controlled atmosphere. The selection task is then transformed from a field operation to a color-sorting line.

The actuation aspect of a sorting line might be a simple tipping mechanism that ejects a fruit from a carrier to fall into an appropriate bin. Alternatively it can take the form of an air-jet that deflects a falling nut kernel, causing it to miss a catching scoop. An essential difference is the substantial delay until a conveyor system reaches the required station compared with the few milliseconds required to actuate a valve to deflect a falling nut.

In all cases there is the task of singulation, separating each item for individual inspection. Apples and tomatoes will roll into cup-shaped depressions, but nut kernels

need to be shaken onto a pair of contra-rotating rollers that form a chute down which the nuts can slide. As the nut falls through the viewing window a decision must be made within milliseconds to either let it fall or to deflect it with the jet.

Some applications hardly warrant the term *machine vision*. In an early nut sorter, light was separated by dichroic mirrors into two chosen wavelengths, to be measured by simple photocells. The transient was interpreted for color and for the presence of shell. A conventional camera would be ineffective, since the delay associated with the frame rate would be long compared with the time of flight. In a tomato sorter, on the other hand, the three conventional red-green-blue (RGB) color bands of a television camera suffice to determine both ripeness and size.

When an early investigation was made of the sorting of broccoli heads, image capture was a substantial hurdle to overcome [46.29]. The reward was the quality of information gathered, enabling grade boundaries to be adjusted when separating the produce for varying destinations. However the project was handicapped by the automation task of separating the heads from bins of produce, to load them into the inspection carriers.

One of the factors used to determine citrus quality is the texture of the skin, which is a combination of wrinkles, dimples and lumps, or flat spots. Previously, measurements were made using a stylus on the revolving fruit [46.30]. In the machine-vision solution, the rotating fruit is illuminated from the side, so that it appears as a *half-moon* to the camera mounted in front of it [46.31]. The *terminator*, dividing lit and portions in shadow, appears as a ragged vertical line, with a statistical distribution of horizontal *roughness* that is readily related to the texture.

Today the vision grading system could well be carried on the harvesting vehicle. There is a growing appreciation of the benefits of single-handling, with grading and packing being performed in the field as part of the picking operation.

46.5 Livestock

46.5.1 Robot Milking

Although matters have come a long way from hand-milking, the milking parlour still requires many operations. The cow must be identified, moved to the milking station, and restrained, then the udder must be

inspected and prepared, and the teat cups attached. The milk must be assessed to ensure that it can safely be added to the storage, the teat cups removed, and the udder treated post-milking.

According to a review paper [46.32] there were over 400 milking robots in operation by 1999, each capable

of tending 40–70 cows by performing these operations. It is now the cow's responsibility to determine the time of milking. The visit to the milking station is rewarded with grain feeding, but training is required to establish a routine.

46.5.2 Sheep Shearing

Without doubt, one of the most spectacular aspects of robotics research for agriculture was the University of Western Australia system for robot shearing, *Shear Magic* [46.33]. A hydraulic robot arm was developed to enable shears to be manipulated in an emulation of the *blows* that are actions of a human shearer. Innovative capacitance sensors enabled the shears to *float* accurately close to and parallel with the skin of the animal, so that nicks and cuts could be reduced below the level a human shearer might impose.

After spending many of millions of dollars, however, the project was discontinued. In an industry where contests are held for speed, the failure of the robot system to achieve similar speeds and cost efficiency left it without support.

A similar fate befell the two-armed system announced in 1985 [46.34]. This was developed for automatically shearing part of the sheep by the Merino Wool Harvesting Pty., which ran out of funding in 1993.

Nevertheless there have been valuable spin-off lessons. An aspect that makes shearing arduous is the need to manhandle the sheep. The robot demanded that the sheep be presented in a structured manner, and the sheep loading animal manipulation platform (SLAMP) sheep-handling part of the university research surely lent

concepts to systems such as *ShearExpress* [46.35, 36]. This is a system whereby the legs of the sheep are cuffed, presenting the sheep at various attitudes for the convenience of manual shearers who can now perform their task standing up rather than crouching, each specializing in a different part of the fleece.

ShearExpress also addresses the problem of bringing sheep and technology into the same physical location, taking the form of a complete mobile shearing shed. Even so, its future is also in question.

There has also been considerable *strategic* payoff from such research. Shearers have moderated their pay demands and industrial relations behavior since the technology was demonstrated. The progress of projects such as these has been hampered by the inability of the industry to gather sufficient investment capital against which to leverage external investment funds.

46.5.3 Slaughtering

Delegates to a Brisbane robotics conference in 1993 were taken on a visit to see a robotic slaughtering system installed at Kilcoy. They were able to compare the manual system, in which each beast is stunned and then hoisted by a hind leg to have its throat cut, against the Fututech system in which the stunning and exsanguination process were automated.

The emotional impact was striking, much more so than the economic factors. The dead animal was toppled onto a moving cradle, where it could be dismantled by human butchers. There were islands of technology, such as the skinning station and the saw that accurately split the carcass in two under sensory control. However the automation was made somewhat cumbersome by the centralized control, requiring extensive cabling to the control and computing booth. The actual saving in manpower was slight, with 30 or so butchers still required for jointing.



Fig. 46.9 *Shear Magic*

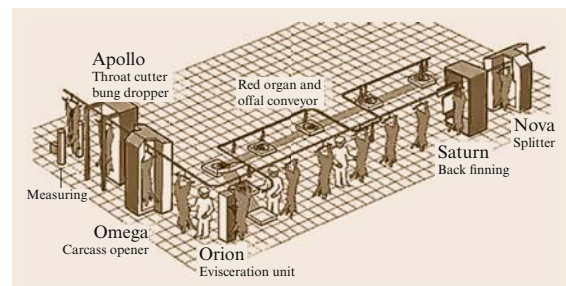


Fig. 46.10 Danish pig line



Fig. 46.11 Sheep with identifying outline

Soon afterwards, the project was wound up. In Hansard on 7 May 1996 [46.37] it was reported that Senator O'Chee said, *What became apparent as the trials went on was that the process was not working. The idea that these animals would be automatically killed just was not happening.* He went on, *That has seen the cost blow out from AU\$2.2 million, which was the estimated cost of the Kilcoy portion of the Fututech program, to \$20 million for that portion of the program. That is half of the \$40 million which was expended in relation to Fututech.*

Perhaps some of the problems lay in overoptimistic overall budget expectations and in the choice of animal to butcher. In Denmark, an automatic slaughter line for pigs claims a throughput rate of 78 000 pigs per week [46.38]. *Islands of automation* handle evisceration and back splitting with methods bearing a similarity to those of Fututech. This part of the US\$270 million DanishCrown Horsens plant was engineered by SFK Meat Systems at a cost of some 20 million dollars [46.39]. Production started in late 2004.

46.5.4 Livestock Inspection

As with produce, grading and classification of livestock carcasses is becoming more commonplace. Poultry [46.40] is one application where the advances in image sensors and filters for particular spectra combine to provide automatic detection of disease and contaminants in a poultry processing line.

An application involving individual recognition of production animals is in full commercial use by *Big Dutchman* [46.41]. In this circumstance, pigs are tagged with unique radiofrequency identification (RFID) ear tags. As each animal enters a feeding bay, the tag is read and an appropriate ration of food is delivered. This allows individualized diets and medication delivery to improve the bottom-line cost/benefit for pork producers.

Another novel method of identifying and controlling animals is being developed in Southern Queens-



Fig. 46.12 Robocow with horse and rider

land [46.42]. Using machine vision, each animal proceeding along a laneway towards water is classified at the species level. An automated gate then either allows or denies access to the watering point. The same technology can be used to remotely draft production animals based on a condition score into several different categories.

46.5.5 Robotic Animals

The Robotic Sheepdog [46.43] was initially developed to herd ducks, with the expectation that it could be applied equally to herding sheep. The robot was controlled by a remote personal computer (PC) using a fixed television (TV) camera to locate both the flock of ducks and the physical robot. The duck behavior was modeled and fed back to the robot position control to herd the animals from one end of an arena to the other.

At the fringes of agriculture can be found country sports such as *cutting contests* and *camp drafting*. Both are contests of horsemanship, requiring a small beast to be manoeuvred by moving the horse relative to the beast. In fact the horse must take many of the decisions. It must be trained to react appropriately to the beast's actions, requiring hours of interaction with a steer or heifer.

The beast is prone to lose patience rather quickly and simply wander away from the horse, so Robocow was developed as a training aid [46.44]. As seen in Fig. 46.12, the steer-shaped body is mounted on a steered tricycle that can spin on the spot or accelerate rapidly on rough ground. Originally programmed to *dance* a chosen pattern, there is a new project to add *bovine intelligence* by enabling it to react directly to the movement of the horse.

46.6 Unmanned Vehicles

There is a rise in the use of unmanned aerial vehicles (UAVs) to address tasks in agriculture. The most advanced applications are already at a commercial stage. Helicopters from Yamaha [46.45] can be programmed to take aerial photographs over a specific flight path. Manual override control allows user interruption, with the capability of resuming the flight path from where it left off. This type of vehicle offers unprecedented availability of a stable platform for image sensors and hyperspectral sensor devices.

UAV collaborative, which has a cooperative research agreement with the National Aeronautical and Space Administration (NASA), uses long duration flight time

UAVs such as the Pathfinder [46.46] to perform unmanned flight operations. The applications range from testing coffee ripeness in Hawaii, to real-time acquisition and distribution of thermal images over a controlled fire in California.

While full-sized tractors may never be fully autonomous, there is scope for smaller, cooperative vehicles to perform set tasks. Many researchers are currently investigating various platforms, including several approaches from researchers in Illinois [46.47]. The idea behind smaller units will be that they can work cooperatively and constantly, thus providing the same amount of horsepower with much reduced risk.

46.7 Conclusions and Future Directions

Robotics is percolating into all aspects of agriculture. The applications are many and various, but there is still great scope for further innovation. A constant grumble is the shortage of manpower for farming, both skilled and for seasonal harvesting operations. Intelligent robotics will be welcome in this most essential industry.

The main challenge facing robotics in forestry is the harsh environment. This is due to a combination of both weather conditions and rapidly changing illumination. Improvements to forest machines are ongoing and some recent results in this direction have been reported by the research project Forestix [46.48, 49].

In agriculture generally, there are a number of issues that must be confronted.

- The ongoing loss of expertise in the industry;
- New robust reliable sensors and actuators will be required to withstand the environmental extremes that are the basic working conditions in many agricultural areas;
- A unit-cost barrier exists due to the small profit margin of many agricultural areas. This will require either a generalized methodology for robotic system, or some other measure to lower unit costs.

References

- 46.1 The ZigBee Alliance. Last access 10 Dezember 2007, from www.zigbee.org
- 46.2 J. Billingsley: Low cost GPS for the autonomous robot farmhand. In: *Mechatronics and Machine Vision*, ed. by J. Billingsley (Research Studies, Baldock 2000) pp. 119–125
- 46.3 A. Halme, K. Hartikainen, K. Kärkkäinen: Terrain adaptive motion and free gait of a six-legged walking machine, *Contr. Eng. Pract.* **2**, 273–279 (1994)
- 46.4 A. Halme, M. Vainio: Forestry robotics – why, what, when. In: *Autonomous Robotic Systems*, ed. by A.T. de Almeida, O. Khatib (Springer, Surrey 1998)
- 46.5 A. Halme, T. Luksch, S. Ylönen: Biomimicking motion control of the WorkPartner robot, *Ind. Robot* **31**, 209–217 (2004)
- 46.6 M. Manninen, A. Halme, R. Myllylä: An aimable laser time-of-flight range finder for rapid scene description, 7th Annu. Conf. Brit. Robot Assoc., Cambridge (1984)
- 46.7 A. Halme, A. Visala, M. Paakkunainen, J. Joensuu, P. Forsman, T. Torvikoski: Model based on-site description of robotized processings, 3rd IFAC/IFIP/IEA/IFORS Conf. Man-Mach. Syst. Anal. Design Eval., Oulu (1988)
- 46.8 P. Viljamaa, H.N. Koivo, A. Peltomaa: Adaptive feed control of a forest harvester, ICOM 2003 Int. Conf. Mechatron., Loughborough University (2003)
- 46.9 T. Högström, A. Wernersson: On segmentation, shape estimation and navigation using 3D laser range measurements of forest scenes, 3rd IFAC Symp. Intell. Auton. Veh., Madrid (1998)
- 46.10 P. Forsman, A. Halme: 3D mapping of natural environments with trees by means of mobile perception, *IEEE Trans. Robot.* **21**, 462–490 (2005)

- 46.11 T. Bailey: *Mobile robot localisation and mapping in extensive outdoor environments*, ed. by Australian Centre for Field Robotics (University of Sydney, Sydney 2002)
- 46.12 C. Brenneke, O. Wulf, B. Wagner: Using 3D laser range data for SLAM in outdoor environments, 2003 IEEE/RSJ Int. Conf. Intell. Robot. Syst., Las Vegas (2003)
- 46.13 J. Guivant, E. Nebot, S. Baiker: Autonomous navigation and map building using laser range sensors in outdoor, J. Robot. Syst. **17**, 565–583 (2000)
- 46.14 S. Thrun, W. Burgard, D. Fox: *Probabilistic Robotics* (MIT Press, Cambridge 2005)
- 46.15 C. Brack: *Forest Measurement and Modeling* (Australian National University, Canberra 2006)
- 46.16 T. Hellström, T. Johansson, O. Ringdahl, F. Georgsson, K. Prorok, U. Sandström: Development of an autonomous path tracking forest machine, Int. Conf. Field Serv. Robot., Port Douglas (2005)
- 46.17 B. Löfgren: Automation – way to increase productivity in logging, NSR Conf. For. Op., Hyytiälä (2004)
- 46.18 P. Kourtz, M. Strome, F. Gougeon: Autonomous forest land-tending robots, Meet. Stat. Methods Math. Comput., Birmensdorf (1992)
- 46.19 J. Billingsley, M. Schoenfisch: Vision-guidance of agricultural vehicles, Auton. Robot. **2**, 65–76 (1995)
- 46.20 NDSU Extension: Center pivot is agricultural production robot. Last access 10 December 2007, from http://www.farmandranchguide.com/articles/2006/02/02/ag_news/production_news/prod23.txt (2006)
- 46.21 D. Elstein, E. Peabody: Fields with a mind of their own?, *Agricult. Res. Mag.* **53**, 14–16 (2005)
- 46.22 C. Perry: Variable-rate irrigation takes guess work out. Last access 10 December 2007 <http://georgiafaces.caes.uga.edu/storypage.cfm?storyid=2930> (2006)
- 46.23 Y. Edan, V. Rogozin: Robotic melon harvesting: Prototype and field tests, *IEEE Trans. Robot. Autom.* **16**, 831–834 (2000)
- 46.24 M. Kanemitsu, K. Yamamoto, Y. Shibano, Y. Goto, M. Suzuki: Development of a Chinese cabbage harvester (Part 1), *Jap. Soc. Agricult. Mach.* **55**, 133–140 (1993)
- 46.25 A. Milella, G. Reina, M. Foglia, A. Gentile: Computer vision applications in agricultural robotics, 11th Conf. Mechatron. Mach. Vis. Pract., Macao (2005)
- 46.26 N. Murakami, K. Inoue, K. Otsuka: Selective harvesting robot for cabbages, Int. Symp. Autom. Robot. Bioproduct. Process. (1995)
- 46.27 M. Slaughter, R. Harre: Color vision in robotic fruit harvesting, *Trans. ASAE* **30**, 1144–1146 (1987)
- 46.28 M. Dunn, J. Billingsley: Machine vision system for counting macadamia nuts, Australasian Conf. Robot. Autom., Brisbane (2003)
- 46.29 J. Billingsley, M. Schoenfisch: Vision and mechatronics applications at the NCEA, Fourth IARP Workshop Robot. Agricult. Food Ind., Toulouse (1995)
- 46.30 R. Leach: *The Measurement of Surface Texture Using Stylus Instruments* (National Physics Laboratory, London 2001)
- 46.31 M. Dunn, J. Billingsley: A machine vision system for surface texture measurements of citrus, Proc. 11th IEEE Conf. Mechatron. Mach. Vis. Pract., Macau (2004)
- 46.32 R.E. Graves: A primer on robotic milking, ASAE/CSAE Annu. Int. Meet., Ottawa (2004)
- 46.33 J. Trevelyan: *Welcome to the Robot Sheep Shearing Project!* Last access 10 December 2007 from <http://www.mech.uwa.edu.au/jpt/shearmagic/Default.html> (2004)
- 46.34 W. A. Scholes: Two-armed sheep-shearing robot on wool firms' threshold Last access 10 December 2007 from <http://www.encyclopedia.com/doc/1G1-3886584.html> (1985)
- 46.35 Shear Express: Shear Express Wool Harvesting offers the key to a new future for the wool industry Last access 10 December 2007 from www.shearexpress.com.au (2006)
- 46.36 J. Trevelyan: *Robots for Shearing Sheep: Shear Magic* (Oxford Univ. Press, Oxford 1992)
- 46.37 Commonwealth of Australia Senate: Official Hansard, Tuesday 7 May 1996. Last access 10 December 2007 from http://parlinfoweb.aph.gov.au/piweb/view_document.aspx?ID=713406&TABLE=HANSARDS (1996)
- 46.38 *Danish Crown Pork Processing Plant*, Horsens Last access 10 December 2007 from www.danishcrown.dk (2006)
- 46.39 SFK Systems: SFK Robots. Last access 10 December 2007 from www.sfk.com (2006)
- 46.40 Y.-R. Chen, K. Chao, M.S. Kim: Machine vision technology for agricultural applications, *Comput. Electron. Agricult.* **36**, 173–191 (2002)
- 46.41 Big Dutchman: Callmatic 2. Last access 10 December 2007 from www.bigdutchman.de/eng (2005)
- 46.42 M. Dunn, J. Billingsley, N. Finch: Machine vision classification of animals. In: *Mechatronics and Machine Vision 2003: Future Trends* (Research Studies, Baldock 2003) pp. 157–163.
- 46.43 R. Vaughan, N. Sumpter, A. Frost, S. Cameron: Robot sheepdog project achieves automatic flock control, Fifth Int. Conf. Simulation Adapt. Behav. (1998)
- 46.44 J. Billingsley, J. Stone, J. Hilton: A mobile robot for training horses, Conf. Field Serv. Robot., Canberra (1997)
- 46.45 Yamaha: *Unmanned Helicopters*. Last access 10 December 2007 from <http://www.yamahamotor.co.jp/global/about/business/sky/index.html> (2006)

- 46.46 UAV Collaborative: *Unmanned Aerial Vehicles*. Last access 10 December 2007 from www.uav-applications.org/projects.html (2006)
- 46.47 T. Grift: U of I creates robot farmers, ACES News. Last access 10 December 2007 from www.aces.uiuc.edu/news/stories/news2813.html (2004)
- 46.48 M. Miettinen, M. Öhman, A. Visala, P. Forsman: Simultaneous localization and mapping for forest harvesters, ICRA'07 2007 IEEE Int. Conf. Robot. Autom., Roma (2007)
- 46.49 M. Öhman, M. Miettinen, K. Kannas, J. Jutila, A. Visala, P. Forsman: Tree measurement and simultaneous localization and mapping system for forest harvesters, 6th Int. Conf. Field Serv. Robot., Chamonix (2007)

Robotics in C

47. Robotics in Construction

Kamel S. Saidi, Jonathan B. O'Brien, Alan M. Lytle

This chapter introduces various construction automation concepts that have been developed over the past few decades and presents examples of construction robots that are in current use (as of 2006) and/or in various stages of research and development. Section 47.1 presents an overview of the construction industry, which includes descriptions of the industry, the types of construction, and the typical construction project. The industry overview also discusses the concept of *automation* versus *robotics* in construction and breaks down the concept of robotics in construction into several levels of autonomy. Section 47.1 also presents traditional and advanced concepts for sensing systems in construction, which enable the use of robots and various forms of automation. Section 47.2 discusses some of the economic aspects of implementing robotics in construction, and Sect. 47.3 presents examples of robots from various construction applications. Section 47.4 discusses unsolved technical problems in construction robotics, which include interoperability, connection systems, tolerances, and power and communications. Finally, Sect. 47.5 discusses future directions in construction robotics and Sect. 47.6 gives some conclusions and suggests resources for further reading.

47.1 Overview	1080
47.1.1 Industry Description	1080
47.1.2 Automation in Construction.....	1081
47.1.3 Robotics in Construction.....	1082
47.1.4 Sensing Systems in Construction.....	1083
47.2 Economic Aspects	1085
47.2.1 Scope	1085
47.2.2 Motivation	1085
47.2.3 Barriers.....	1085
47.3 Applications	1086
47.3.1 Construction	1086
47.3.2 Maintenance and Decommissioning	1091
47.4 Currently Unsolved Technical Problems	1093
47.4.1 Interoperability.....	1093
47.4.2 Structural Connection Systems	1093
47.4.3 Tolerances	1094
47.4.4 Power and Communications in the Field	1095
47.5 Future Directions	1095
47.6 Conclusions and Further Reading	1096
References	1096

Construction is a ubiquitous human activity that relates to the creation or realization of physical artifacts or custom-made capital goods. It is distinguished from manufacturing in that the production activity normally occurs in a field setting and is undertaken in the open air, on natural terrain, and often with naturally occurring materials. Typically, building and construction products are large in scale and unique in form. Moreover, the environment or field setting is typically unique and requires a, rather ad hoc, *factory* to be synthesized on site.

Over the centuries, various forms of machines and mechanical engineering systems have been intro-

duced into the construction engineering domain and into the building and construction industry to increase production efficiency. In common with the fields of agriculture, mining, and forestry, the long-term trend has been for these fields to become increasingly mechanized [47.1].

In the last few decades, with the decrease in the relative cost of machinery to labor and with the globalization of markets, the construction industry has become significantly more capital intensive and large-scale machinery systems and pieces of construction plant – such as tunnel-boring machines and very large tower

cranes – have become commonplace. This trend to mechanization is likely to continue with the progressive introduction of computer-controlled construction machinery and flexible manufacturing concepts into the industry.

47.1 Overview

The application of robots in construction traditionally falls under *construction automation*. As the term implies, the field of construction automation is focused on automating construction processes, and the use of robots is but one aspect of automation. Construction processes also fall within several categories best described through a brief introduction to the construction industry.

47.1.1 Industry Description

In the USA the value of construction put in place in 2002 was approximately 8% of the national gross domestic product (GDP) [47.2], employing over 7 million workers in 2005 [47.3]. According to data from a 2003 US Census Bureau report, approximately half (48.25%) of those workers were employed by firms of four employees or fewer, and roughly 60% of all construction workers worked for specialty trade contractors who accounted for roughly 60% of construction firms [47.4] (specialty trade contractors are usually subcontractors on a project and are not responsible for the overall outcome of the project). In the European Union (EU), construction accounted for 5.7% of the EU’s gross value added in 2004, and the construction industry employed over 11 million workers [47.5]. Worldwide, construction industry *spending* is estimated at approximately 11% of the world’s GPD [47.6].

Construction is considered by many to be technologically behind other industries, such as manufacturing. In manufacturing, a product is designed for mass production, whereas construction *products* (or projects) are

usually one-off and unique [47.7]. Thus the efficiencies achieved through mass production are not easily achieved in construction.

Other often attributed reasons for the construction industry’s technological lag are the industry’s fragmentation and aversion to the risks associated with the introduction of new technologies [47.7, 8]. In addition, unlike their manufacturing counterparts, construction sites are for the most part unstructured, cluttered, and congested, making them difficult environments for robots to operate in. Furthermore, human workers are also present in large numbers on a construction project, making safety a paramount concern.

Types of Construction

Construction projects are usually classified as residential, commercial, industrial, or civil. Residential construction generally involves single-family homes or large apartment buildings; commercial focuses on building structures such as office and retail space, warehouses, and so on; industrial is involved in building factories, power plants, and other similar structures; and civil construction focuses on public infrastructure such as highways, bridges, tunnels, and dams.

In the USA (as of 2006), residential construction accounted for a little over half of the total construction value put in place [47.2]. Among nonresidential construction activities, commercial, civil, and industrial projects accounted for approximately 26%, 17%, and 4%, respectively, according to census data for the 4 years between 2002 and 2005 [47.2]. According to the same census data, approximately 78% of all construction activity in the USA is undertaken by private firms.

In the EU (as of 2005), civil construction accounted for 20% of the total construction put in place, nonresidential construction for 31%, renovation and maintenance for 23%, and new residential building for 26% [47.9].

The Typical Construction Project

A construction project typically goes through six major phases, as shown in Fig. 47.1. Some projects may

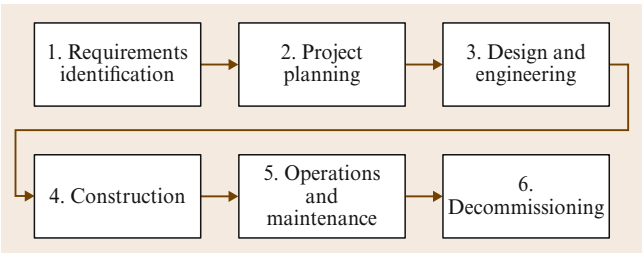


Fig. 47.1 Phases of a typical construction project

go through different variations of the sequence shown, but most projects include planning, design, construction, and operation phases [47.4,7]. Phase 1 begins when a need for a project arises and the requirements are identified. Phase 2 involves developing alternative project plans that could meet the identified needs and evaluating the technological and economic feasibility of each alternative. Phase 3 develops detailed engineering designs and specifications for the plan selected in phase 2. The construction of the facility from ground-breaking through to final inspection takes place in phase 4 of the project. The facility is occupied and commences operation in phase 5 and continues operating until it is time to shut down and dismantle the facility once it becomes obsolete (phase 6).

The actual physical work (building, operating, maintaining, and dismantling) on a construction project takes place in the last three phases. Although these are the only phases in which machines are used, construction automation can also take place during the other phases of a project, as described in Sect. 47.1.2.

During the various phases of a construction project, several stakeholders may be involved at any time. The major stakeholders include the following: the *owner* and *operator*, whose needs initiated the project; the *architect* and *engineer*, who have the task of translating the owner's needs into an aesthetically pleasing and structurally sound design; and the *general contractor*, whose task is to translate the design into a physical structure. In addition several other stakeholders may be involved either independently from the above stakeholders or as part of their organizations; for example, the constructor may employ a project manager, a construction manager, a site superintendent, and others, and he/she will typically subcontract major activities such as excavation and concrete pouring. The subcontractors and sub-stakeholders are often the ones who use a new technology and can either *make or break* its implementation.

Although the builder is the most likely user of robots on a construction project, the actual work on site is often conducted by subcontractors who often are reluctant or financially unable to use advanced technologies which have not been entirely adopted by industry. A constructor's goal is often to meet the owner's requirements by the most efficient and least risky methods possible. Hence, traditional construction methods that have stood the test of time are preferred. Nevertheless, it has often been anecdotally reported that the owner has *the power* to require the use of certain technologies on a construction project, since that shifts part of the risk and cost associated with the use of the technology to the

owner. This phenomenon may be partially responsible for the widespread penetration of laser scanning technology into the construction industry that is currently taking place [47.10].

47.1.2 Automation in Construction

Construction automation describes the field of research and development focused on automating construction processes, and the use of robots is but one aspect of that field. In short, construction automation deals with applying the principles of industrial automation to the construction sector, whether in building construction, civil engineering (roadways, dams, bridges, etc.), or in prefabrication of construction components [47.11]. This can be viewed as an extension to research in field service robots generally designed to replace or assist humans in a specific construction-related task or function.

From a historical perspective, research in construction robotics and automation started in the 1980s with the introduction of single-purpose robots (principally remotely controlled, or teleoperated, machines). The Japanese led this effort, driven primarily out of concern for societal demographics, which showed a significant future shortfall in personnel available for the construction labor pool [47.12]. In the US the principal related work involved developing remote control or teleoperated machinery for hazardous work that required modified construction equipment. Example applications include robots developed for rapid runway repair and unexploded ordinance removal. In the EU, research was focused on the development of large-size masonry (brick laying, assembly) robots for residential and industrial building construction.

During the next decade, as research on task-specific construction robots continued, large Japanese construction firms introduced on-site *factories* for high-rise construction. These construction systems included just-in-time delivery of components, automated part tracking and material handling, robotic connection and assembly, and centralized control in an enclosed or semi-enclosed environment. It is reported that the systems enable better working conditions (weather invariant) and reduced project completion time [47.13]. Other benefits include improved productivity and quality, though overall construction costs are not necessarily lower [47.14].

Advanced concepts in integrated residential construction automation were developed as part of the EU FutureHome project. In this construction concept, each structure consists of several high-quality, prefabricated three-dimensional (3-D) modules and two-dimensional

(2-D) panels which are assembled *production-style* on site [47.15]. An analogue of this approach for automated residential construction has been commercialized in Japan [47.16].

New methods for collecting, processing, analyzing, and communicating construction information are a significant area of construction automation research [47.17]. This research includes data interoperability and exchange through the design, construction, operations, maintenance, and decommission phases of a capital project [47.18]; advanced sensors for assessing the status of the construction process [47.19–21]; visualization systems for planning construction events, verifying constructability, and maintaining site situational awareness [47.22–24]; and information models which extend traditional computer-aided design (CAD) modeling to combine both the physical (geometric) and functional characteristics of building components [47.25].

Finally, three notable, large-scale, ongoing EU projects are attempting to integrate construction automation into the way construction projects are executed: the ManuBuild (open building manufacturing) project deals with, among other things, the development of mobile field factories, including robots, for on-site modular construction [47.26]; the I3CON (industrialized, integrated, intelligent, construction) project deals with the indoor automation and robotization of buildings [47.27]; and the Tunconstruct project deals with the robotization of inspection and maintenance operations in tunnels [47.28].

47.1.3 Robotics in Construction

Construction robotics is an advanced form of mechanization (automation) in which an endeavor is made to automate some industrially important operation and thereby reduce the cost of this operation by either removing a human operator *from the control loop*, or enhance operational efficiency through machine control systems. Due to the nature of construction work, most robots which have been developed for the construction industry are either mobile or relocatable systems. Some platforms, such as floor-finishing robots and machine-controlled earthmovers require mobility as a specific function of the work process to be performed. Others, such as wall and ceiling panel manipulators, require some level of mobility to extend their operating workspace.

Robot terminology can vary depending upon the research discipline. For this chapter three general cat-

egories of construction robots are introduced. The first class is *teleoperated systems*, which for simplicity includes *remote control systems*. The general distinction between the two terms is whether or not the equipment must be operated in line of sight from the human controller [47.29]. The second category, *programmable construction machines*, includes most construction equipment that is outfitted with sensors and mechanisms to augment operation by an onboard human operator. The final category, *intelligent systems*, relates to unmanned construction robots which operate either in a semi- or fully autonomous mode. In Sect. 47.3 this classification will be expanded to include examples based on various generic activities, materials handled (operand types), levels of onboard intelligence, levels of commercialization, and levels of system integration and computer integration.

Teleoperated Systems in Construction

In established engineering terminology, the term *teleoperation* refers to the remote control of machines and systems. In teleoperation (loosely referred to as telerobotics) the control of the machine is accomplished by the use of remote control means such as an umbilical cord or wireless control. Teleoperation ideas and methods are used extensively in the space and nuclear industries (Chaps. 45 and 48, respectively).

In telerobotics, the machine does not operate autonomously but is under the control of a human. Data sensing and interpretation and cognitive activities such as task planning are done by the operator.

Recently, many telerobotic devices have appeared in the construction and mining industries. These machines have evolved in response to industrial situations where there is danger to the operator and where remote-controlled machinery is necessary (e.g., teleoperated small compactors). Situations of this kind occur in the construction, demolition, and mining industries and in other hazardous locations.

The technology for telerobotics in construction is well established, with a number of excellent examples of such activity available, for example, a sophisticated model-based supervisory-control-type distributed teleoperation system for the construction of a trench for a diversion dam in a lava field in Japan using a fleet of heavy earthmoving machines [47.30].

Programmable Construction Machines

A software-programmable construction machine is what most people would consider to be a robot. The operator of this type of machine is able to vary the task to be ac-

complished within certain constraints either by choosing from a preprogrammed menu of functions or by *teaching* the machine a new function. Variations in the task to be accomplished could be as simple as slight changes in the driving speed based on the current load for an automated forklift, or as complex as a change from being able to pick and place steel beams and columns using an autonomous crane to being able to deliver concrete autonomously using the same crane.

Generally, software-programmable construction machines are identical to traditional construction machines (such as an excavator), but have been modified to be controllable through a computer [similar to the way in which traditional manufacturing machines – such as mills and lathes – evolved into computer numerically controlled (CNC) machines].

Software-programmable construction machines can make use of an electronic representation of a portion of the construction site where their work is to be conducted in order to control all or part of the machine's operation. A commercialized example is *stakeless grading*, where data from a 3-D model is used in combination with global positioning systems (GPS) and/or laser measurement systems to automate the blade control for bulldozers and motor-graders.

Intelligent Systems in Construction

As opposed to a teleoperated or a software-programmable construction machine a fully autonomous construction robot is expected to accomplish its task, within a defined scope, without human intervention. A semiautonomous construction robot would be expected to accomplish its task with some level of planning interaction conducted with a human supervisor [47.29]. In each case the construction robot is expected to adapt to its sensed environment, formulate plans for the execution of its task, and replan as necessary (with possibly some human assistance in the semiautonomous mode). The intelligent construction robot should also be able to determine when its tasking is not executable and request assistance.

Example research in intelligent construction systems includes autonomous excavation [47.31, 32] and autonomous crane operations [47.24, 33–35].

47.1.4 Sensing Systems in Construction

Field measurements are an integral part of the construction process. The *tape measure* and the *transit* have been used in construction for decades for measuring distances and angles, respectively. However, construction meas-

urements are not limited to distances and angles, but can also include measurements of installed quantities, percentages of completion of activities, and so on. All of these measurements are necessary to be able to lay out the site where a facility is to be built, to measure the conformance of the as-built facility to the intended design, and to monitor safety, productivity, and progress.

Traditional Sensing Systems

In addition to the tape measure, transit, and level, all of which are still in use [47.36], more modern sensing systems are now considered commonplace (or traditional) at a construction site. These include total stations [or digital theodolites – a combination of a theodolite and an electronic distance measuring (EDM) device], GPS, and laser planes, levels, and plummets. Although GPS is commonplace in construction (and many other industries) and may be considered a traditional sensing system at this stage, it is and will continue to be a significant enabler for construction automation.

Advanced Sensing Systems

More advanced sensing systems are also beginning to have some impact in the construction industry. Some of these technologies are, and will become more so in the future, key enablers for the introduction of automation and robotics onto the construction job site by providing real-time situational awareness of site conditions. Some of these advanced sensing systems are described below.

Real-Time Positioning Systems. Traditional GPS relies on line of sight (LOS) to Earth-orbiting satellites and cannot currently track a moving object at much better than 20 cm unless ground-based stations are used. For applications where LOS to the GPS satellites is not available (such as indoors) or where higher real-time accuracy is required, two technologies currently exist: laser-based (also often referred to as indoor GPS) and ultra-wideband positioning systems. Indoor GPS (iGPS) relies on LOS to a series of laser transmitters (beacons) to measure and track the position of an optical receiver at up to 100 m away from the closest beacon [47.37]. Ultra-wideband (UWB) relies on LOS to a series of radio receivers to measure and track the position of a radio transmitter (working in the microwave frequency) at up to 200 m away from the closest receiver. Both technologies are similar to satellite-based GPS in that multiple objects can be tracked using the same infrastructure created by a few fixed beacons or receivers and that tracking can be quickly regained following a break in the LOS. (In the case of UWB, direct LOS is not always necessary

since the **UWB** signals can travel through many common construction materials, albeit at the possible cost of degradation in the positioning accuracy.)

Laser-Based 3-D Imaging Systems. Laser-based three-dimensional (3-D) imaging systems use lasers to measure the range to points on objects within their field of view, as well as the bearing (angles) to those points within the instrument's coordinate frame. These systems usually output a *cloud* of points (or *point clouds*) with 3-D coordinates (usually *X*, *Y*, *Z* or angle, angle, range) associated with each point. In addition, many of these systems also output the intensity of the returned signal from each measured point on the objects as well as the red, green, and blue (**RGB**) color information from each point. Laser-based 3-D imaging systems are commonly also referred to by some of the following names: laser scanners, laser radar or laser detection and ranging (**LADARs**), and light detection and ranging (**LIDARs**).

The use of laser-based 3-D imaging systems is rapidly penetrating the construction industry due to its efficiency in capturing the existing conditions of a construction project (during or after construction) in great detail [47.10]. Gathering as-built conditions is essential for facility redesign, maintenance, repair, and upgrade and the ability to efficiently and accurately measure the condition of the existing facility and can have significant time and cost savings.

Ground-Penetrating Radar. Ground-penetrating radar (**GPR**) uses electromagnetic radiation at the microwave frequency (also known as *ultra-wideband*) range to measure reflections from objects buried underground or within structures (similar to traditional **RADAR**). In construction, this technology has been used to locate buried utilities before digging [47.38], to measure pavement layer thickness [47.39], to inspect concrete bridge decks [47.40], and to measure soil water content [47.41].

Structural Health Monitoring Systems. Structural health monitoring technology uses in situ, nondestructive sensors [e.g., strain gages, ultrasonic sensors, optical fibers, micro-electromechanical systems (**MEMS**) sensors] in order to detect structural damage or dangerous conditions [47.42]. These sensors are best deployed as a network of wireless sensors and can be used to monitor the state of a structure as it is being built, to monitor its health during normal operation, and to help determine structural integrity after an extreme event.

Equipment Health Monitoring Systems. Similar to structural health monitoring, these systems allow equipment operators to monitor the state of the machine (e.g., peak loads, tire pressures, engine temperature, hours in operation, crane tilt). The systems can also suggest when a machine needs maintenance and can even safely shutdown a machine should dangerous or undesirable conditions occur.

Concrete Maturity Meters. Concrete maturity meters use disposable sensors that are embedded in the concrete before it is poured in order to measure the internal temperature of the curing concrete and determine the developed strength [47.43]. Other systems use the dielectric properties of wet concrete to estimate the developed strength [47.44].

Radiofrequency Identification. Radiofrequency identification (**RFID**) is a relatively new addition to the construction sector. With the complete acceptance and adoption of **RFID** by some of the world's largest retailers (such as Walmart and Target) and by the **US** Department of Defense, this technology is evolving rapidly and the construction industry is poised to benefit from it.

The implementation of **RFID** technology involves the combination of **RFID** tags and tag interrogators. Each **RFID** tag has a unique identification number permanently stored on a microchip [usually in read-only memory (**ROM**)] embedded within the tag. Tags can be passive, semiactive (or semipassive), or active. An interrogator transmits a radio wave at a specific radiofrequency (**RF**) and waits for returned signals from tags that are within range. For passive tags, an onboard capacitor accumulates enough of the interrogator's **RF** energy (due to current induced in the tags antenna) to transmit the tag's unique identifier back. For active tags, an onboard battery powers the transmit function and can be used to maintain data stored in random-access (volatile) memory (**RAM**). In semiactive tags, an onboard battery is used to maintain data stored in **RAM**, but the transmit function still relies on the induced current to charge a capacitor.

Applications of **RFID** technology in construction to date have been in the research and development domain. Pilot projects have been conducted to test the effectiveness of **RFID** for tracking the delivery of construction components (e.g., pipe spools) to the construction site and to maintain an inventory of available parts and tools on site [47.45–48]. The application of **RFID** technology is likely to benefit the construction industry significantly in the short term (since projects are often plagued

47.2 Economic Aspects

47.2.1 Scope

In general, construction system designers are interested in the development of efficient production systems using available technology. Typically, such systems designers use new processes and technologies as they become available and as they prove to be economically viable. In relation to automation technology, such technology will not be introduced if only *islands of automation* are implemented with little overall system benefit [47.49].

47.2.2 Motivation

Apart from the pursuit of economic efficiency, construction robotics has technical features that have the potential to add value to the construction process and to product timeliness and quality, for example, there are work quality benefits such as in the quality and reliability of welding and inspection tasks. Furthermore, automated performance monitoring systems can give construction agencies hard-copy data for quality control and quality assurance purposes. The provision of historical compaction data in the construction of roads is a good case in point [47.50].

As a unique technical feature, construction robotics has the potential for reducing of occupation health and safety (OH&S) risks by removing workers from danger zones and high levels of noise or air pollution [47.51]. Construction can be a dangerous activity with a relatively large number of people being killed or injured annually (when compared to other industries).

Robotics has a major potential for allowing construction work to be undertaken in very cold or very hot weather and in areas subject to extreme radiant heat (such as building or bush fires or near blast furnaces) or nuclear radiation, for example, an early use of tele-robotics was the remote control of concrete pumps in the encasement of the Chernobyl nuclear power plant in the Ukraine. Likewise, the remote control of underwater bulldozers and other types of underwater equipment is an obvious area for the technical attributes of construction robotics.

47.2.3 Barriers

Technical and Commercial Barriers

While technology researchers and developers may be interested in construction robots for their own sake,

the typical purchaser of such systems is a contractor or construction agency who is predominantly interested in machinery systems as an instrument of purpose or as a *tool-of-the-trade*. For a particular application, robotic or automated systems or machines must compete with existing labor-intensive solutions and alternative technological solutions.

As a generalization in 2006, it is fair to say that currently available construction robotic solutions do not undercut traditional labor-based solutions except in special niche areas. This is in contrast to the widespread use of robotics in the manufacturing sector.

Some of the economic difficulties associated with the application of robotics in construction stem from the characteristics of the industry. Because it is primarily concerned with the production of large-scale custom-made capital goods that are specifically designed to fit a site, the construction industry is characterized by unique products and low production volumes and in situ development processes in open-air environments. Historically, the construction industry has proven substantially unsuited to the hard-automation methods that have been successfully developed in the large-volume repetitive applications in the manufacturing sector [47.52].

Using manufacturing industry terminology, construction may be categorized as a *jobbing-type* industry. Typically, the construction industry has very low levels of repetition and normally has *job lots of one*.

To increase the level of internal repetition, methods of product standardization have been introduced. Over the last 50 years, standardization and extensive experimentation with systems-building methods have occurred in the industrialized building sector (in places such as the old Union of Soviet Socialist Republics) and in attempts to produce manufactured homes and buildings. Such attempts have largely failed due to a lack of product and process flexibility and to market acceptance of the products [47.53].

Potentially, the emergence of robotics and flexible manufacturing methods will have a great impact on the construction industry since product and process variety can be accommodated much more easily than with hard-automation methods.

Because the construction industry is fully established and has its own methods and techniques, the likelihood is that it will robotize its own forms of equipment rather than simply move industrial robots from the factory to the site.

Sociological and Industrial Barriers

Apart from commercial barriers many people are concerned that there will be substantial industrial or worker

resistance to the widespread introduction of on-site robotics. The possible reason for worker resistance is potential job losses through displacement of labor.

47.3 Applications

Applications of robots in construction span a wide and diverse range. While certain robots with limited intelligence are slowly finding their way onto construction sites around the world, others remain scaled-down prototypes under development at universities and research organizations.

Classifying the extent of construction robot applications using a single scheme would be a daunting task, especially since some robots are being developed for multiple applications. Classification could be by level of commercial success, type of task performed, industry sector, physical scale of device, degree of technical complexity, level of onboard intelligence, and so on. Classification could also be by type of actual construction activity involved.

From a purely physical-world and practical point of view, construction may be viewed as being comprised of a finite number of elementary processes [47.55–58], which may be summarized by the following list [47.8]:

- Attaching
- Building
- Coating
- Concreting
- Connecting
- Covering
- Cutting
- Digging
- Finishing
- Inlaying
- Inspecting
- Jointing
- Measuring
- Placing
- Planning
- Positioning
- Spraying
- Spreading

Most of these processes can also be grouped into three predominant types of functional operators as follows:

1. materials handling (by bulk and unit load)
2. materials shaping (cutting, breaking, compacting, and machining)
3. structural joining

These functional operators are typically each applied to multiple operands. Common operands in building and civil engineering are steel and other metals, concrete, timber, earth and rock, masonry, plastic and glass, cement, aggregate, epoxy resin, bitumen, and other bulk and formed materials.

In the interest of brevity, the applications presented in Tables 47.1–47.10 have been loosely organized into two categories: (1) construction and (2) maintenance and decommissioning. Within these categories, robots are organized by the type of construction activity that each performs, but not necessarily following any of the classification schemes presented above. The list of applications is not comprehensive, but is meant to be a fairly representative sample of the types of robots currently being used in, or developed for, the global construction industry. (Although certain commercial construction equipment are included in this section, the inclusion of such information should in no way be construed as indicating that such products are endorsed by the National Institute of Standards and Technology (NIST) or are recommended by NIST or that they are necessarily the best equipment for the purposes described.)

47.3.1 Construction

Table 47.1 Materials handling (a) Six degree of freedom robotic crane [47.54]

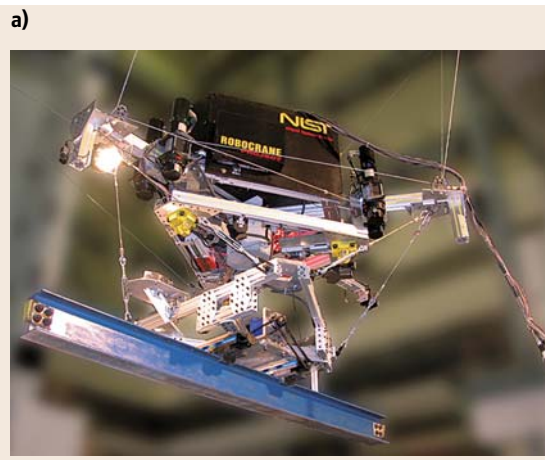


Table 47.1 (b) Concrete panel installation robot (courtesy Fujita Research) **(c)** Large scale pipe manipulator **(d)** Large manipulator system (courtesy Shimizu Corp.)



Table 47.2 Earthmoving **(a)** Teleoperated excavation system (courtesy Fujita Research) **(b)** Computer assisted road compacting system [47.59] **(c)** Automated grading system (courtesy Caterpillar, Inc.)



Table 47.2 (d) Automated drag line control system [47.60] (e) Autonomous excavator robot [47.61] (f) Autonomous off-road dump truck [47.62]

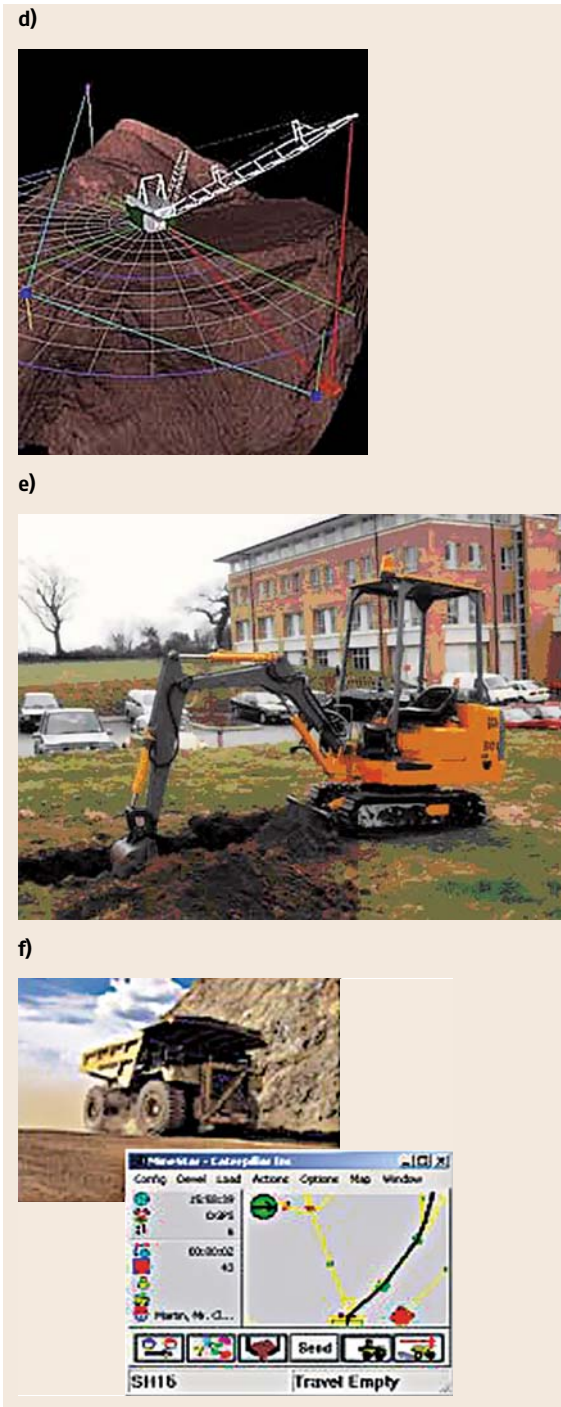


Table 47.3 Concrete placement and finishing (a) Concrete surface finishing robot [47.63, 64] (b) Programmable, articulated-boom machine for the pumped delivery of fresh concrete [47.65] (c) Auto-leveling concrete screed machine (courtesy Somero Enterprises, Inc.)

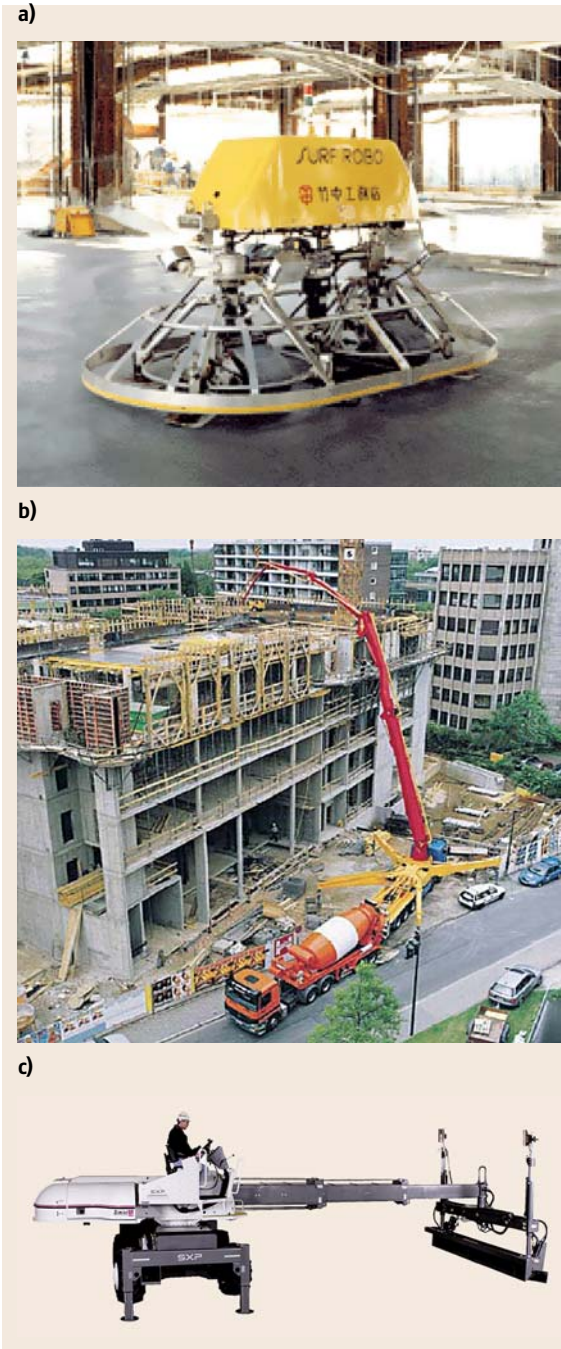


Table 47.3 (d) Concrete slipform road paving machine (courtesy GOMACO Corp.) **(e)** Teleoperated concrete spraying robot (courtesy MEYCO Equipment) **(f)** Concrete block laying robot [47.66]

d)



e)



f)

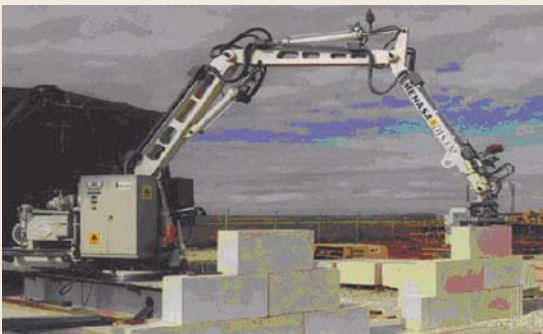


Table 47.4 Hard and soft rock tunneling Automated rock drilling robot (courtesy Bever Control AS)



Table 47.5 Precast concrete production **(a)** Automated concrete pipe manufacturing machines (courtesy Hawkeye Group USA)



Table 47.5 (b) Precast concrete handling robot (courtesy Halo, Inc.)



Table 47.6a,b Structural steelwork production and erection (a) Autonomous steel column welding robot [47.11] (b) Autonomous structural steel pick-and-place robot [47.67]

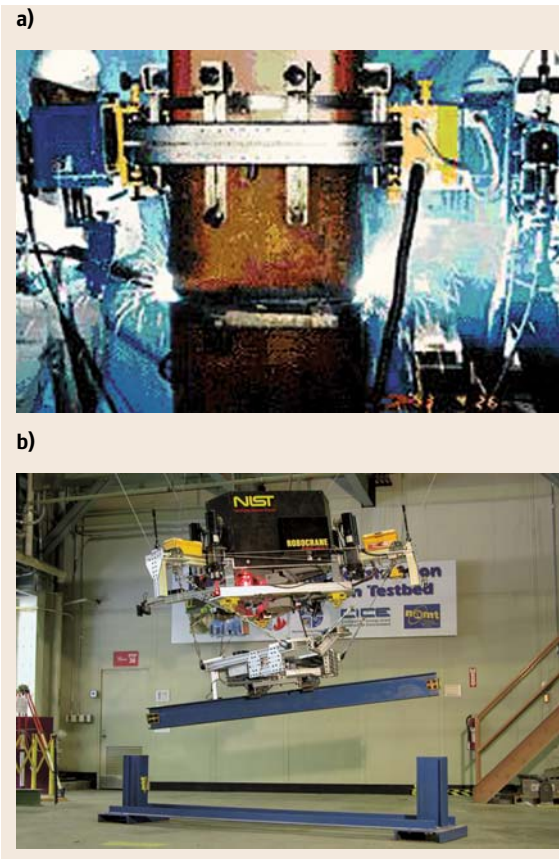
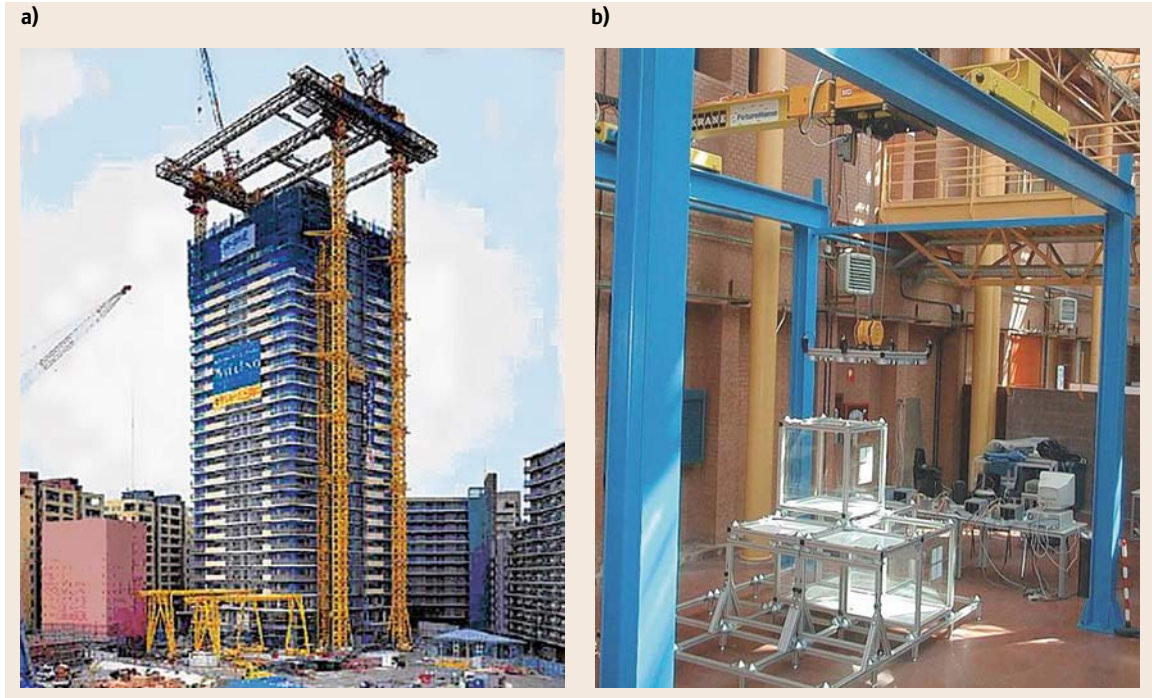


Table 47.6 (c) Teleoperated rebar placing robot [47.11]



Table 47.7a,b Whole building production systems (a) Onsite factory building construction [47.68] (b) Autonomous manufactured home construction robot [47.15]



47.3.2 Maintenance and Decommissioning

Table 47.8 Road maintenance (a) Teleoperated pothole patching robot (courtesy Leeboy)

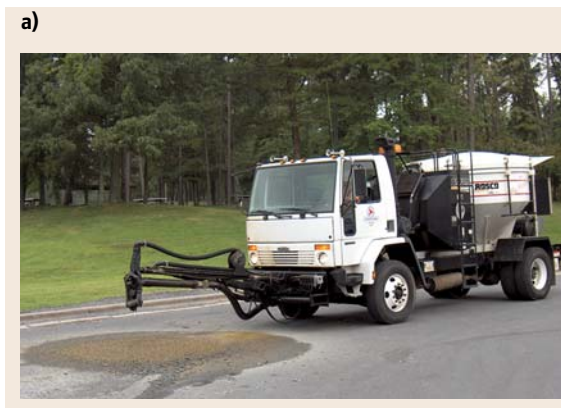


Table 47.8 Road maintenance (b) Semi-autonomous road crack sealing robot [47.69]



Table 47.8 Road maintenance (c) Semi-autonomous road crack sealing robot [47.70]



Table 47.9 Inspection (a) Teleoperated climbing inspection robots [47.71]



Table 47.9 Inspection (b) Teleoperated bridge maintenance and inspection robot [47.72]

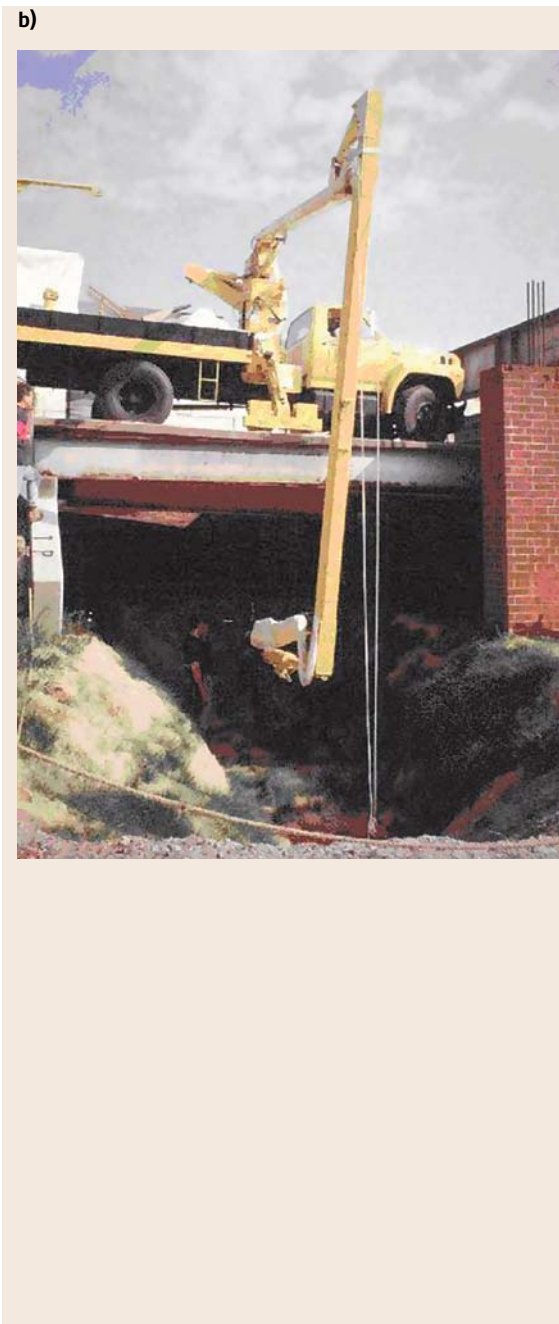
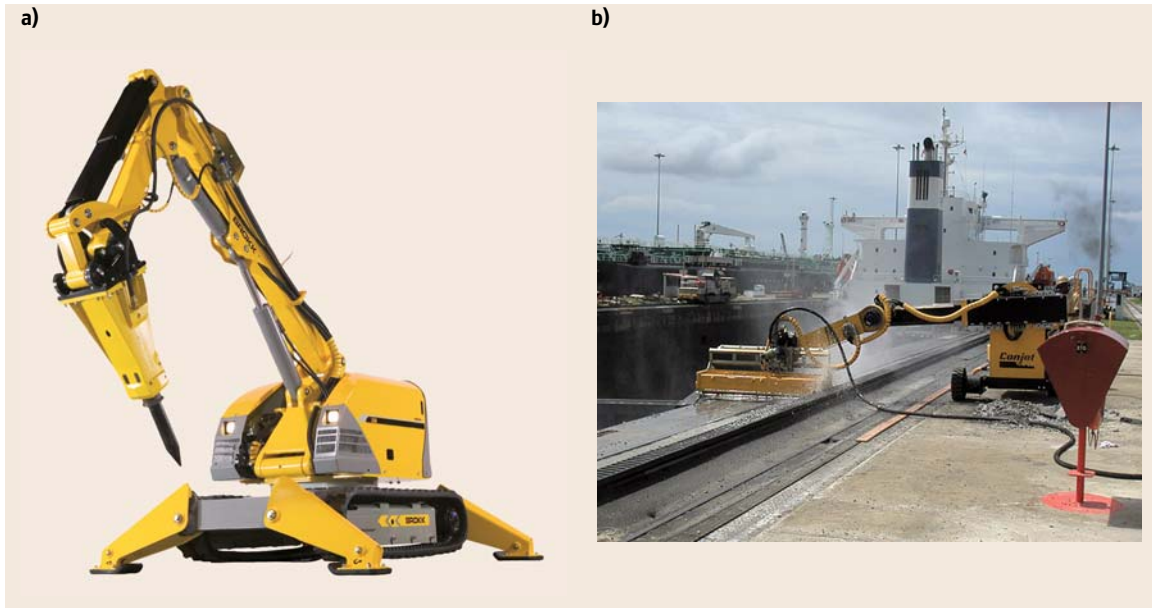


Table 47.10a,b Demolition (a) Teleoperated demolition robot (courtesy Brokk AB) (b) Water-jet demolition robot (courtesy Conjet AB)



47.4 Currently Unsolved Technical Problems

47.4.1 Interoperability

A recent study found that the US construction industry does not realize approximately 15 billion US dollars per year in potential savings due to inadequate interoperability related to information exchange and management practices [47.73]. Although the lack of interoperability between the various information systems used in construction is a significant source of inefficiency for the industry, it is also a roadblock to the use of automated systems in construction. Automated systems need electronic information on past, current, and/or projected future states of a construction project to function efficiently.

For example, in order for a robotic crane to pick a steel beam from the site and deliver it to its target location, the robot must be able to know that the steel beam has been delivered to the site, as well as its current position and orientation. While information on the current inventory of parts on a site may be available on paper, it is rarely available electronically unless someone enters it manually into some computer system, which in

turn may not be compatible with other systems used in that project. In many of the examples of automated construction technologies presented in this chapter, custom electronic databases and/or data formats have to be devised to demonstrate the robot's operation. In some cases, the electronic information had to be entered manually from paper into a computer for the robot to work correctly.

In addition to information exchange and management, many of the measurement instruments and sensors used in construction are not interoperable. This problem is not limited to the construction sector but is a relatively large problem in many robotic applications where different types of sensors are used. Several efforts are underway to make sensors interoperable [47.74] and construction equipment in general [47.75], but the issues have yet to be resolved.

47.4.2 Structural Connection Systems

Traditionally, structural member connections in construction have been designed for human installation.

Whether using bolted, welded, or other types of connections, manual labor is usually involved in guiding the mating parts together and in establishing the connection.

For example, in structural steel erection, workers perched on the structure typically guide a crane operator through visual or auditory cues in order to maneuver a steel beam (or column) into place. The workers must then physically manipulate the beam in order to align corresponding surfaces for bolting or welding. Once the correct beam pose has been achieved, it must be maintained while the workers temporarily fasten the beam to the structure. The workers then release the beam from the crane and permanently fasten the beam to the structure at a later stage.

For automated (or robotic) construction to work, new connections that are more amenable to automation must be designed. These connections need not mimic traditional, human-installed connections, but should be optimized for use with robots instead. For example, the Lehigh University Advanced Technology for Large Structural Systems (ATLSS) Center designed a gravity-load-only shear steel connector [47.76] back in the early 1990s that is more suitable for automated construction (Fig. 47.2). This type of male–female connector for automatic assembly of building modules allows, together with an adequate control strategy, small tolerances, which permit assembly by automatic cranes [47.77]. An example of a nonstructural connector that is more amenable to automation is the piping and electrical connector developed for the FutureHome project, shown in Fig. 47.3 [47.15].

Although automated welding has been applied in some limited form in construction, it is generally used to replace manual welding without changing the design of the parts being welded. In other words, no significant change in the way in which construction components are designed has occurred as a result of automated welding. Moreover, apart from a few examples of automated welding applied at the construction site, most of the other limited applications are done at the component fabrication facility.

47.4.3 Tolerances

Specifications for tolerances in the construction sector exist for most types of construction, however, they are not always achieved in practice [47.78]. For example, it has been stated that one of the biggest sources of problems in structural steel erection in the US is that fabricated pieces are often out of tolerance, and that this is only discovered during installation at the construc-



Fig. 47.2 A drop-in, shear-load-only, steel connection

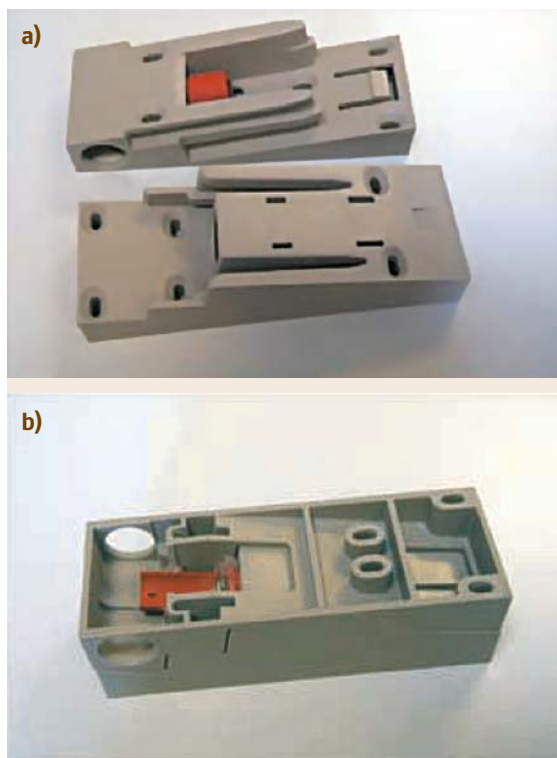


Fig. 47.3a,b A piping and electrical drop-in connector shown (a) disassembled and (b) assembled

tion site [47.79]. However, since the finished facility must meet the design tolerances before it is accepted, this shifts the burden to workers and supervisors dur-

ing construction. The workers are expected to handle problems as they come up rather than expect all fabricated construction components to be within tolerance. Since most construction projects are under tight schedules, it is often preferable to fix these kinds of problems on site rather than wait for replacement components to be fabricated and delivered.

However, tolerance problems are not all due to fabrication errors. Installation problems are also responsible for out-of-tolerance problems, for example, anchor rod installation has been an area of concern for structural steel erection. (Anchor rods (or bolts) are the connection interface between the concrete foundation and the structural steel columns. The rods are usually installed by the concrete foundation crew before the concrete is dry.) The situation of anchor rod patterns that do not match the hole patterns in a mating column is an identified problem in construction [47.20, 80].

Given the relatively loose achievable tolerances in construction, the application of robotics in construction faces an uphill battle. This, in addition to the unstructured nature of the construction site environment, requires that robots either be highly intelligent in order to correctly interpret and react to their surroundings or to be human assisted. However, site structure and tolerances are expected to improve as pressures to reduce costs and improve productivity continue to rise. Improvements in site organization and construction tolerances have already been proved to be achievable in a few cases, as

has been demonstrated in Japan [47.53]. However, the economic case for these demonstration projects has yet to be made.

47.4.4 Power and Communications in the Field

Unlike manufacturing environments, in which specially designed factories are outfitted with the necessary power and communications installations, a construction project often begins before such resources have been installed at the site. Therefore, robots with large power requirements that need to communicate with supervisory systems located off site would be challenging to implement without significant added cost.

Although communications technologies have advanced significantly in the last few decades, it is still considered difficult to maintain a reliable local-area network at a construction site and to connect that network to the Internet. The use of cellular telephones with a press-to-talk (the digital version of the traditional two-way radio communication method) feature has largely replaced the traditional two-way radios that also revolutionized on-site construction communications. However, in order for construction sites to become more automated, reliable interference-free high-bandwidth networks must be able to carry data transmission between sensors, machines, and supervisory systems.

47.5 Future Directions

As previously discussed, developing new methods for collecting, processing, analyzing, and communicating construction information is a significant area of construction automation research, and will dominate near-term efforts. As this construction information becomes more readily accessible, automation of processes can be enabled directly from the combination of design information and current site status that is accurately captured and shared. Resource tracking will become ubiquitous and just-in-time delivery of needed materials and equipment will happen throughout the site.

Advances in mobility (humanoid robotics, smart cars, legged locomotion, etc.) will enable ever more

automated material handling on the job site. These advances will require better control systems for the construction robots that can provide high payload and good positioning accuracy. The increased use of robots at the construction site will also drive research into safety systems for construction robots working around human workers and other machines.

Perhaps most important, more extensive automatic design systems will enable more prefabrication of building components and new methods of assembling those components on site, which in turn will provide the promise of faster, better, and cheaper construction robotics first envisioned in the 1980s.

47.6 Conclusions and Further Reading

The application of robotics to construction has yet to catch up with other industries such as automobile manufacturing. Construction presents a unique challenge for robotic applications. The construction environment is cluttered, unstructured, and teaming with human workers. In addition, construction processes are usually labor intensive and have to accommodate wide margins of error in the constructed facility. The application of robotics in construction to date has been limited to commercial teleoperated and programmable machines. Autonomous or semiautonomous machines are currently mostly limited to research projects within various nonconstruction organizations. With the increase in competition throughout the global construction market, construction companies are on the lookout for ways to improve productivity, quality, and safety. The use of automation and robotics is one answer that the industry is slowly turning toward. However, before these potential solutions can be successfully applied, much work is needed to improve construction tolerances, develop

standards, and achieve real-time site status monitoring.

The International Association for Automation and Robotics in Construction holds an annual conference (the International Symposium on Automation and Robotics in Construction) at which researchers can present the latest developments in the field. The proceedings from this conference hold a wealth of information on the state of the art of the field and are accessible to the general public through IAARC's website (www.iaarc.org).

There are several journals that publish articles on various aspects of automation in construction. Most notable of these are *Automation in Construction*, *Computer-Aided Civil and Infrastructure Engineering*, the *Journal of Computing in Civil Engineering*, and the *Journal of Construction Engineering and Management*. In addition, some journals publish special editions on robotics in construction. Some examples are *Autonomous Robots* and the *Journal of Advanced Robotic Systems*.

References

- 47.1 E. Ginzberg: The mechanization of work, *Sci. Am.* **247**(3), 66–75 (1982)
- 47.2 US Bureau of the Census: *Value of Construction Put in Place – Seasonally Adjusted Annual Rate* (Luscensus, Washington 2006), available at: <http://www.census.gov/>
- 47.3 Bureau of Labor Statistic: *Industry at a Glance, NAICS 23: Construction* (US Department of Labor, Washington 2006), available at: <http://www.bls.gov/iag/construction.htm>
- 47.4 US Bureau of the Census: *Statistics of U.S. Businesses* (Luscensus, Washington 2006), available at: <http://www.census.gov/csd/susb/>
- 47.5 European Communities: *Europe in Figures – Eurostat Yearbook 2005* (European Commission, Brussels, 2005), available at: http://epp.eurostat.ec.europa.eu/cache/ITY_OFFPUB/KS-CD-05-001/EN/KS-CD-05-001-EN.PDF
- 47.6 D. Crosthwaite: The global construction market: a cross-sectional analysis, *Constr. Man. Econ.* **18**(5), 619–627 (2000)
- 47.7 D.W. Halpin, R.W. Woodhead: *Construction Management*, 2nd edn. (Wiley, New York 1998)
- 47.8 K.S. Saidi: Possible Applications of Handheld Computers to Quantity Surveying, Diss. (The Univ. Texas, Austin 2002)
- 47.9 Eurostat: *European Business – Facts and Figures* (European Commission, Brussels 2007), , available at: http://epp.eurostat.ec.europa.eu/pls/portal/url/page/PGP_MISCELLANEOUS/PGE_DOC_DETAIL?p_product_code=KS-BW-06-001
- 47.10 T. Greaves, B. Jenkins: *Capturing Existing Conditions with Terrestrial Laser Scanning: a Report on Opportunities, Challenges and Best Practices for Owners, Operators, Engineering/Construction Contractors and Surveyors of Built Assets and Civil Infrastructure* (Spar Point Research, Danvers 2004)
- 47.11 L. Cousineau, N. Miura: *Construction Robots: the Search for New Building Technology in Japan* (ASCE, Reston 1998)
- 47.12 J.G. Everett, H. Saito: Construction automation: demands and satisfiers in the United States and Japan, *J. Constr. Eng. Man. ASCE* **122**(2), 147–151 (1996)
- 47.13 M. Taylor, S. Wamuziri, I. Smith: Automated construction in Japan, *Proc. Inst. Civil Eng. Civil Eng.* **156**(1), 34–41 (2003)
- 47.14 J. Maeda: Current research and development and approach to future automated construction in Japan, *Proc. Congress Construction Research Congress* (2005) p. 2403
- 47.15 C. Balaguer, M. Abderrahim, J.M. Navarro, S. Boudjabeur, P. Aromaa, K. Kahkonen, S. Slavenburg, D. Seward, T. Bock, R. Wing, B. Atkin: FutureHome: an integrated construction automation approach, *IEEE Robot. Autom. Mag.* **9**(1), 55–66 (2002)

- 47.16 Y. Maruyama, Y. Iwase, K. Koga, J. Yagi, H. Takada, N. Sunaga, S. Nishigaki, T. Ito, K. Tamaki: Development of virtual and real-field construction management systems in innovative, intelligent field factory, *Autom. Constr.* **9**(5–6), 503–514 (2000)
- 47.17 C. Balaguer: Soft robotics concept in construction industry, *World Automation Congress* (2004) pp. 517–522
- 47.18 K.A. Reed: The Role of the CIMSteel Integration Standards in Automating the Erection and Surveying of Structural Steelwork, 19th Int. Symp. Autom. Robot. Constr. SP989 (NIST, Gaithersburg, 2002)
- 47.19 N.J. Shih: The application of a 3-D scanner in the representation of building construction site, *ISARC 2002: 19th Int. Symp. Autom. Robot. Constr.* (2002) pp. 337–342
- 47.20 B. Akinci, F. Boukamp, C. Gordon, D. Huber, C. Lyons, K. Park: A Formalism for utilization of sensor systems and integrated project models for active construction quality control, *Autom. Constr.* **15**(2), 124–138 (2006)
- 47.21 G.S. Cheok, W.C. Stone: Non-intrusive scanning technology for construction assessment, *IAARC/IFAC/IEEE. Int. Symp.* (1999) pp. 645–650
- 47.22 K. McKinney, M. Fischer: Generating, evaluating and visualizing construction schedules with CAD tools, *Autom. Constr.* **7**(6), 433–447 (1998)
- 47.23 B. Akinci, M. Fischer, J. Kunz: Automated generation of work spaces required by construction activities, *J. Constr. Eng. Manage. ASCE* **128**(4), 306–315 (2002)
- 47.24 V. Kamat, R. Lipman: Evaluation of standard product models for supporting automated erection of structural steelwork, *Autom. Constr.* **16**(2), 232–241 (2006)
- 47.25 C.M. Eastman: *Building Product Models* (CRC, Boca Raton 1999)
- 47.26 M. Vega, T. Bock, A. Malone: *Newsletter 3* (ManuBuild Open Building Manufacturing, Moorgate 2006), available at: <http://www.manubuild.org/>
- 47.27 G. Aouad, J. Kirkham, P. Brandon, F. Brown, G. Cooper, S. Ford, R. Oxman, M. Sarshar, B. Young: Information modeling in the construction industry – the information engineering approach, *Constr. Manage. Econ.* **11**(5), 384–397 (1993)
- 47.28 G. Beer: Tunconstruct: a new european initiative, *Tunnels Tunneling Int. FEV* (2006) pp. 21–23
- 47.29 H.M. Huang: Autonomy Levels for Unmanned Systems (ALFUS) Framework Volume I: Terminology Version 1.1, NIST SP1011 (NIST, Gaithersburg 2004), available at: http://www.isd.mel.nist.gov/projects/autonomy_levels/publications/NISTSP_1011_ver_1.1.pdf
- 47.30 Y.F. Ho, H. Masuda, H. Oda, L.W. Stark: Distributed control for tele-operations, *IEEE/ASME Trans. Mechatron.* **5**(2), 100–109 (2000)
- 47.31 S. Singh: State of the art in automation of earth-moving, *ASCE J. Aerosp. Eng.* **10**(4), 179–188 (2002)
- 47.32 H. Quang, M. Santos, N. Quang, D. Rye, H. Durrant-Whyte: Robotic excavation in construction automation, *IEEE Robot. Autom. Mag.* **9**(1), 20–28 (2002)
- 47.33 J. Albus, R. Bostelman, N. Dagalakis: The NIST RoboCrane, *J. Robot. Syst.* **10**(5), 709–724 (1993)
- 47.34 K.S. Saidi, A.M. Lytle, W.C. Stone, N.A. Scott: Developments toward automated construction (NIST Interagency Rep., 7264 2005)
- 47.35 S.C. Kang, E. Miranda: Physics based model for simulating the dynamics of tower cranes, 10th Int. Conf. Comput. Civil Building Eng. (ICCCBE) (Weimar 2004)
- 47.36 C. Lindfors, P. Chang, W. Stone: Survey of construction metrology options for AEC industry, *J. Aerosp. Eng.* **12**, 58 (1999)
- 47.37 S. Kang, D. Tesar: A Novel 6-DoF measurement tool with indoor GPS for metrology and calibration of modular reconfigurable robots, *IEEE ICM Int. Conf. Mechatron.* (Istanbul 2004)
- 47.38 L.E. Bernold, L. Venkatesan, S. Suvarna: Equipment mounted multi-sensory system to locate pipes, *Pipelines* **130**, 112 (2004)
- 47.39 D.A. Willett, K.C. Mahboub, F. PE, B. Rister: Accuracy of ground-penetrating radar for pavement-layer thickness analysis, *J. Transp. Eng.* **132**, 96 (2006)
- 47.40 C.L. Barnes, J.F. Trottier: Effectiveness of ground penetrating radar in predicting deck repair quantities, *J. Infrastruct. Syst.* **10**, 69 (2004)
- 47.41 J.A. Huisman, S.S. Hubbard, J.D. Redman, A.P. Annan: Measuring soil water content with ground penetrating radar – a review, *Vadose Zone J.* **2**(4), 476–491 (2003)
- 47.42 G.W. Housner, L.A. Bergman, T.K. Caughey, A.G. Chassiakos, R.O. Claus, S.F. Masri, R.E. Skelton, T.T. Soong, B.F. Spencer, J.T.P. Yao: Structural control: past, present, and future, *J. Eng. Mech.* **123**(9), 897–971 (1997)
- 47.43 United States Department of Transportation: *Maturity Meters: A Concrete Success*, ed. by L. Pope (Federal Highway Administration (FHWA), Washington 2002)
- 47.44 S.V. Ramaiah, B.F. McCullough, T. Dossey: Estimating in Situ Strength of Concrete Pavements Under Various Field Conditions (The Univ. of Texas, Austin 2001) Center Transport. Res.
- 47.45 J. Song, C. Haas, C. Caldas, E. Ergen, B. Akinci, C.R. Wood, J. Wade: *Field Trials of RFID Technology for Tracking Fabricated Pipe – Phase II* (FIATECH, Austin 2003), available at: <http://www.fiatech.org>
- 47.46 J. Song, C.T. Haas, C. Caldas, E. Ergen, B. Akinci: Automating the task of tracking the delivery and receipt of fabricated pipe spools in industrial projects, *Autom. Constr.* **15**(2), 166–177 (2006)

- 47.47 J. Aksoy, I. Chan, K. Guidry, J. Jones, C.R. Wood: *Materials and Asset Tracking Using RFID: A Preparatory Field Pilot Study* (FIATECH, Austin 2004), available at: <http://www.fiatech.org>
- 47.48 J. Kang, P. Woods, J. Nam, C.R. Wood: *Field Tests of RFID Technology for Construction Tool Management* (FIATECH, Austin 2005), available at: <http://www.fiatech.org>
- 47.49 J.B. O'Brien: Holistic automation of construction projects, 13th Int. Symp. Autom. Robot. Constr. (1996) pp.179–188
- 47.50 F. Peyret, J. Jurasz, A. Carrel, E. Zekri, B. Gorham: The computer integrated road construction project, Autom. Constr. **9**(5), 447–461 (2000)
- 47.51 J.B. O'Brien: Robotics as a technology for risk reduction in construction work, First Annual Risk Engineering Seminar – Implementation of Performance based risk and safety standards (1996) pp. 60–72
- 47.52 R.H. Hayes, S.C. Wheelwright: *Restoring Our Competitive Edge: Competing Through Manufacturing* (Wiley, New York 1984)
- 47.53 Y. Miyatake: SMART system: A full-scale implementation of computer integrated construction, 10th Int. Symp. Autom. Robot. Constr. (Houston 1993)
- 47.54 A.M. Lytle, K.S. Saidi: NIST research in autonomous construction, Auton. Robots **22**(3), 211–221 (2007)
- 47.55 J.G. Everett, A.H. Slocum: Automation and robotics opportunities – construction versus manufacturing, J. Constr. Eng. Manage. ASCE **120**(2), 443–451 (1994)
- 47.56 L.A. Demsetz: Task identification for construction automation, 6th Int. Symp. Autom. Robot. Constr. (1989) pp. 95–102
- 47.57 R. Kangari, D.W. Halpin: Potential robotics utilization in construction, J. Constr. Eng. Manage. **115**(1), 126–143 (1989)
- 47.58 R.L. Tucker: High payoff areas for automation applications, 6th Int. Symp. Autom. Robot. Constr. (1988) pp. 9–16
- 47.59 F. Peyret: The Achievements of the computer integrated road construction project, 17th IAARC/CIB/IEEE/IFAC/IFR Int. Symp. Autom. Robot. Constr. (IS-ARC) (Taipei 2000)
- 47.60 Commonwealth Scientific, Industrial Research Organisation: *Mining Robotics Project* (CSIRO, Clayton South 2006), available at: <http://www.csiro.au/csiro/content/standard/ps1kk.html>
- 47.61 D.A. Bradley, D.W. Seward: The development, control and operation of an autonomous robotic excavator, J. Intell. Robot. Syst. **21**(1), 73–97 (1998)
- 47.62 Pacific Coal, C. Hughes: *Project C8001: Introduction of Autonomous Haul Trucks. Final Report* (Australian Coal Research, Brisbane 1997)
- 47.63 K. Kikuchi, S. Furuta, T. Imai: Development and the result of practical works of concrete floor finishing robot, 5th Int. Symp. Autom. Robot. Constr. (1988)
- 47.64 K. Arai, B. Yamada, M. Saito, K. Banno: The development and practice test of the Mark II mobile robot for concrete floor slab finishing, 5th Int. Symp. Autom. Robot. Constr. (1988)
- 47.65 H. Benckert: Mechydronic for boom control on truck-mounted concrete pumps, Tech. Symp. Constr. Equipment Technol. 2003 (Dresden 2003)
- 47.66 E. Gambao, C. Balaguer, F. Gebhart: Robot assembly system for computer-integrated construction, Autom. Constr. **9**(5–6), 479–487 (2000)
- 47.67 K.S. Saidi, A.M. Lytle, W.C. Stone, N.A. Scott, N.A. Scott: Developments toward automated construction (NIST Interagency Rep. 7264, 2005)
- 47.68 Y. Miyatake, Y. Yamazaki, R. Kangari: The SMART system project: a strategy for the management of information and automation technology in computer integrated construction. In: *Management of Information Technology for Construction*, ed. by S. Marthur Krishan (Worls Scientific, Brighton 1993) pp. 407–420
- 47.69 C. Haas, K. Saidi, Y. Cho, W. Fagerlund, H. Kim, Y. Kim: Implementation of an Automated Road Maintenance Machine (ARMM) (Univ. of Texas, Austin 1999) Center for Transportation Research, Project Summary Report
- 47.70 D.A. Bennett, X. Feng, S.A.V.A. Maintenance, C. Technology: AHMCT Automated crack sealing program and the operator controlled crack sealing machine, Transportation Res. Board 2003 Annual Meeting (2003)
- 47.71 C. Balaguer, A. Gimenez, J.M. Pastor, V.M. Padron, M. Abderrahim: A climbing autonomous robot for inspection applications in 3-D complex environments, Robotica **18**(03), 287–297 (2000)
- 47.72 S.J. Lorenc, B.E. Handlon, L.E. Bernold: Development of a robotic bridge maintenance system, Autom. Constr. **9**(3), 251–258 (2000)
- 47.73 M.P. Gallaher, R.E. Chapman: Cost Analysis of Inadequate Interoperability in the US Capital Facilities Industry (National Institute of Standards and Technology, Gaithersburg 2004) US Dept. of Commerce, Technology Administration
- 47.74 K.B. Lee, M.E. Reichardt: Open standards for homeland security sensor networks, Instrum. Meas. Mag. IEEE **8**(5), 14–21 (2005)
- 47.75 E.F. Begley, M.E. Palmer, K.A. Reed: Semantic mapping between IAI ifcXML and FIATECH AEX models for centrifugal pumps (National Institute of Standards and Technology, Gaithersburg, 2005)
- 47.76 R. Fleischman, B.V. Viscomi, L.W. Lu: Development, analysis and experimentation of ATLSS connections for automated construction. Proceedings of the 1st World Conference on Steel Structures (Acapulco 1992)
- 47.77 S. Garrido, M. Abderrahim, A. Gimenez, C. Balaguer: Anti-swinging input shaping control of an automatic construction crane, IEEE Trans. Autom. Sci. Eng. (2007)

- 47.78 J.K. Latta: *Inaccuracies in Construction* (National Research Council Canada, Ottawa 1975), available at: http://irc.nrc-cnrc.gc.ca/pubs/cbd/cbd171_e.html
- 47.79 A.M. Lytle, K.S. Saidi: Application of laser-based 3-D imaging to steel construction, ISARG (2006)
- 47.80 A.M. Lytle, K.S. Saidi: Proceedings of the Automated Steel Construction Workshop June 6-7, 2002 (National Institute of Standards and Technology, Gaithersburg 2004), NIST Internal Report 7108

48. Robotics in Hazardous Applications

James P. Trevelyan, Sung-Chul Kang, William R. Hamel

Robotics researchers have worked hard to realize a long-awaited vision: machines carrying people from burning buildings or tunneling through collapsed rock falls to reach trapped miners. In this chapter we review progress. Researchers still have many challenges ahead of them but there has been remarkable progress in some areas. Hazardous environments present special challenges for the accomplishment of desired tasks depending on the nature and magnitude of the hazards. Hazards may be present in the form of radiological or toxicity dangers to potential explosions. Technology that specialized engineering companies can develop and sell without active help from researchers marks the frontier of feasibility. Just inside this border lie teleoperated robots for explosive ordnance disposal (EOD) and for underwater engineering work. Even with the typical tenfold reduction in manipulation performance imposed by the limits of today's *telepresence* and teleoperation technology, robots usually offer a more cost-effective solution. Most hazardous applications lie far beyond the frontier, although researchers managed to establish some limited inroads by the turn of the 21st century. Fire fighting, rescue oper-

- 48.1 Operation in Hazardous Environments: The Need for a Robotics Solution..... 1101
- 48.2 Applications..... 1103
 - 48.2.1 Eradicating Landmines 1103
 - 48.2.2 Hazardous Materials Handling and Operations 1110
- 48.3 Enabling Technologies 1114
 - 48.3.1 Mobility Issues..... 1114
 - 48.3.2 Manipulator Design and Control for Hazardous Object Handling..... 1117
 - 48.3.3 Control for Hazardous Tasks..... 1117
 - 48.3.4 Data Communications..... 1120
 - 48.3.5 Energetics 1120
 - 48.3.6 System Architectures for Real-Time Mission Control 1121
- 48.4 Conclusions and Further Reading 1122
- References 1123

ations, removing high-level nuclear contamination, reactor decommissioning, tunneling through rock falls, and most landmine and unexploded ordnance problems still present many unsolved problems.

48.1 Operation in Hazardous Environments: The Need for a Robotics Solution

Hazardous environments present special challenges for the accomplishment of desired tasks depending on the nature and magnitude of the hazards. Hazards may be present in the form of radiological or toxicity dangers to potential explosions. When the magnitudes of hazards reach the point that human exposure would either represent a direct threat to life or long-term health consequences, some form of remote operations that separate humans from the hazards must be employed. An

extensive example of such operations is environments involving nuclear radiation; in fact many of the technical roots of modern robotics technology can be traced back to nuclear remote handling manipulators and support systems. Remote handling and operations concepts using engineered systems that allow humans to work effectively from a safe environment have evolved over the years. Today, such remotely operated systems are used widely in many areas and recently they have become rou-

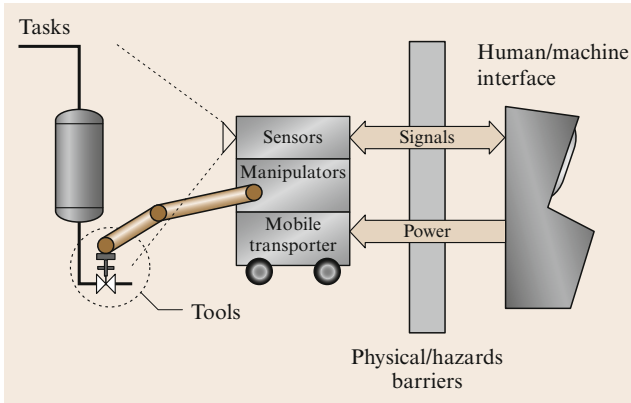


Fig. 48.1 Basic subsystems of a remote handling system

tinely used in explosives disposal, security operations, and handling of dangerous biological materials.

A remote handling system will generally involve subsystems for mobility, manipulation, tooling, sensing, and human-machine interfacing (Fig. 48.1). Nominal operations involve the collective workings of these subsystems to accomplish remote operational goals. Any remote handling system will eventually experience some aspect of off-nominal operation that may be the result of unexpected environmental events or system malfunctions. The fundamental idea is to connect the human operator to the remote environment via a power and signal infrastructure that allows effective operation in

the remote environment: mechanized devices and sensor systems that allow the human's perception and action capabilities to be *projected* into the hazardous environment to perform remote operations. The more realistic this projection, the more natural and effective the human remote control will be.

Because of their inherent complexity and the very nature of remote operations, operator training is a major challenge that requires the use of simulations and cold testing facilities that provide operators with comprehensive and realistic training. Such training will typically encompass all aspects of nominal and anticipated off-nominal operations.

Remote handling systems themselves will eventually experience equipment failures. Remote maintenance/operation of failed remote maintenance systems must be an integral part of their basic design and operational features. Hardware/software features must be provided for the analysis of, recovery from, and correction of problems.

Robotics researchers have worked hard to realize a long-awaited vision: machines carrying people from burning buildings or tunneling through collapsed rock falls to reach trapped miners. In this chapter we review progress. Researchers still have many challenges ahead of them but there has been remarkable progress in some areas.

Technology that specialized engineering companies can develop and sell without active help from researchers

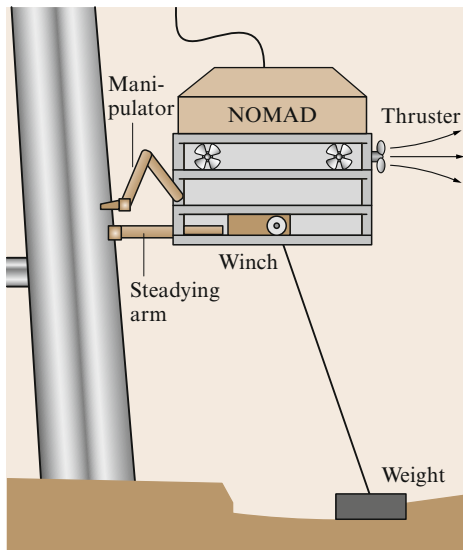


Fig. 48.2 NOMAD remotely operated vehicle for underwater engineering work (Total Marine Systems Pty Ltd, Fremantle, Australia)

marks the frontier of feasibility. Just inside this border lie teleoperated robots for explosive ordnance disposal (EOD) [48.1, 2] and for underwater engineering work. The necessity for people to wear protective suits in these situations limits their endurance and dexterity. Even with the typical tenfold reduction in manipulation performance imposed by the limits of today's *telepresence* and teleoperation technology, robots usually offer a more cost-effective solution. Figure 48.2 illustrates the elegant simplicity of the NOMAD remotely operated vehicle (ROV) that pulls itself deeper with a simple electric winch. It can be left in place overnight or during rough weather. Free-swimming robots are much more complicated and expensive and have to be hauled out of the water when not under active control.

Limited autonomy or autonomous operation for a restricted time can relieve operator fatigue and allows

unmanned aerial vehicles (UAVs) to fly extended reconnaissance missions with occasional precision weapon delivery. Such missions would be too hazardous or too politically sensitive for manned aircraft.

Most hazardous applications lie far beyond the frontier, although researchers managed to establish some limited inroads by the turn of the 21st century. Fire fighting, rescue operations, removing high-level nuclear contamination [48.3], reactor decommissioning, tunneling through rock falls, and most landmine and unexploded ordnance problems still present many unsolved problems. Attempts to use first-generation mine rescue robots reported in the press in 2000 and 2001 merely created distractions for the people who regularly have to risk their lives saving their trapped colleagues (e.g., the Australian Numbat robot [48.4]).

48.2 Applications

Applications of robotic systems in hazardous environments encompass an extremely wide spectrum. The solutions for these different environments are equally diverse. In general such applications involve unique challenges associated with the uncertainty and unstructured nature of the associated tasks. In this discussion, two application areas that are very different have been selected to give the reader a deeper sense of technology evolution, accomplishments, and remaining challenges. Landmine eradication, sometimes referred to as demining, is a current application domain with humanitarian importance and extremely difficult and dangerous outdoor conditions. Hazardous materials and operations is a decades-old problem area that has had a strong influence on many aspects of robot manipulation and mobility research and development.

48.2.1 Eradicating Landmines

Efforts to construct a practical robotic device to help with landmine clearance have met with only limited success. While remote control technology has enabled some existing machines and vehicles to be used in hazardous situations, we are still a long way from achieving reliable robotic mine clearance. It is instructive to understand why many expectations have turned out to be hopelessly optimistic.

Landmines, a simple type of victim-activated explosive device, were used extensively in Europe and North

Africa during the Second World War between 1939 and 1945. Extensive clearance operations in 1945 and 1946 removed nearly all of the landmines then in use ([48.5], part 1, pp. 15–25).

Landmines were used extensively in subsequent decades. Along with antipersonnel cluster bombs, they caused extensive civilian and military casualties in Vietnam and Cambodia from the 1960s onwards. However, it was not until their widespread use in Afghanistan, Angola, Cambodia, and several other countries in the 1980s that they were recognized as a major humanitarian problem. Landmines blocked aid efforts needed to rebuild communities following civil conflicts.

The Red Cross and the International Campaign to Ban Landmines (ICBL) successfully promoted a ban on the use of landmines that came into effect in 1997 as the Ottawa Treaty. The knowledge that thousands of children were losing their legs, even their lives, motivated hundreds of researchers to develop new technologies to help eliminate this threat. By 2000 ICBL estimated that over 80 countries were affected by landmines and other explosive remnants of war such as cluster bombs. Although many countries that have not signed the Ottawa Treaty still have extensive stocks of landmines, the treaty has been effective in restricting the use of landmines as much through peer pressure as enforcement. A few countries and several nonstate actors continue to deploy landmines according to recent reports. However, explosive remnants of war such as cluster bombs and

other munitions have become, yet again, an increasing problem in Iraq and Afghanistan.

There are several basic types of landmines and unexploded ordnance (UXO) that continue to cause problems in many countries.

Antipersonnel (AP) blast mines, made predominantly of plastic with small metal firing pins and detonator cases, typically contain between 20 g and 100 g of explosive (Fig. 48.3). These mines only cause extensive injuries when they detonate within a few centimetres of a person. Typically they lie buried just below the ground surface and are activated when the victim steps on top of the mine. Shattered fragments of bone pass through the flesh of the leg at high velocity. If the victim survives long enough to reach hospital, amputation above or below the knee usually saves his or her life but the victim will need prosthetic legs, replaced at regular intervals.

Antipersonnel fragmentation mines contain similar quantities of explosive with a thick metal case that breaks into high-velocity fragments when the mine explodes just above the ground surface. Older fragmentation mines were mounted on posts; more recent varieties lie buried but jump into the air when activated and explode at waist height, killing or seriously wounding victims up to 200 m away. These mines are much easier to de-

tect so they are often protected by nearby antipersonnel blast mines to deter theft.

Antivehicle (AV) or antitank mines (AT), made predominantly of plastic with small metal firing pins and detonator cases, are large versions of the AP blast mine and typically contain 5–10 kg of explosive. Some have a thick metal plate on top that can penetrate 50 cm of armor plate on the underside of a tank. These mines cause significant damage even to mine-resistant vehicles which have blast-resistant hulls, offset wheels, and additional protection for occupants.

Air-dispersed munitions such as cluster bombs (CBs) were not intended to be victim-activated. Several hundred are released from a single canister and they are designed to explode on impact with the ground. Typically between 5% and 25% failed to explode immediately and lie in a partially triggered state either on or just below the ground surface. Some will detonate in response to electromagnetic fields from metal detectors, and others will detonate with the slightest movement. Most explode like powerful fragmentation mines with a lethal radius of up to 200 m.

Improvised explosive devices (IEDs), often in the form of roadside bombs, are increasingly used by insurgent groups fighting organized military forces (asymmetric warfare). They are often made from large UXOs fitted with remote controlled detonators. Ironically the UXO is often unintentionally donated by the same organized military forces who become the targets of these devices.

Evolution of Landmine Clearance Techniques

Removing landmines is difficult. It is important to distinguish between humanitarian mine clearance and military mine clearance methods (sometimes called *breaching*). Military mine clearance has to work fast, in all conditions (even under fire), and therefore it is unrealistic to aim for 100% clearance. In humanitarian operations there is less time pressure and work can be suspended in unfavorable conditions, and the aim is 100% clearance. Recent political expectations of low casualties often demand very high clearance standards even in military operations.

Humanitarian mine clearance typically starts years, perhaps decades, after the mines were laid. The mines lie buried or hidden from view. They deter people from entering the land so vegetation often grows thickly. Drainage systems rapidly become clogged, denying access in wet conditions.

The traditional *manual* method for removing landmines has been to use a metal detector to locate metal

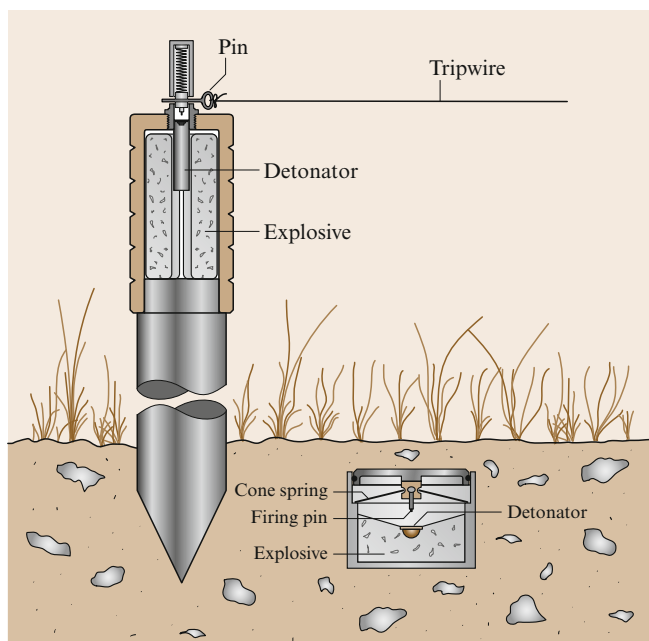


Fig. 48.3 Antipersonnel fragmentation mine (left) and buried blast mine (right)



Fig. 48.4 Typical ruined house overgrown by vegetation in a village in northern Croatia, possibly containing mines or booby traps. The entire village population was forced to leave in 1991 and the houses were looted and intentionally severely damaged. Vegetation problems like this must be taken into account in considering practical mine and **UXO** clearance devices. August 1999 (photo J. Trevelyan)

fragments close to the ground surface and then to carefully check each metal fragment to see if it is associated with a mine or explosive device. Any tripwires and vegetation have to be removed, with great care, before a metal detector can be used. In many areas deminers have to investigate hundreds or thousands of metal fragments for every mine found. Manual mine clearance also requires careful organization and marking of the ground

to ensure safety and thorough clearance. Currently it is still the method that guarantees the lowest risk of residual mine contamination but it is expensive, typically costing **US\$1–\$5/m²**.

Armored mine clearance machines using hammers mounted on the end of rapidly spinning chains (flails) first appeared in the 1940s but have not been able to neutralize mines with sufficient reliability for most humanitarian applications [48.6].

In the late 1990s commercial mine clearance organizations operating in thick vegetation in Bosnia Herzegovina and Croatia realized that flails spinning just above the ground could rapidly remove vegetation and trip wires to prepare the ground for manual clearance, often assisted by mine detection dogs. Clearance costs have been reduced by up to 80% (particularly in thick vegetation) using different combinations of machines, detection dogs, and manual clearance.

Ground milling machines use metal drums studded with hard cutters that shred buried objects. They require more power than flails but can operate with greater levels of reliability. Both flails and ground milling machines have been extensively used in Croatia to recover large areas of formerly productive agricultural land. Both kinds of machines can withstand a limited number of **AT** and moderate-size **UXO** explosions before main bearings and other components need to be replaced.

Naturally, machines operate best on flat or gently sloping ground, which is also the land that is most valu-



Fig. 48.5 Flail machine using hammers on the ends of spinning chains to clear vegetation and tripwires. This machine will also detonate a proportion of buried mines (*inset*) (photos Scanjack AB, Sweden)

able for agriculture and human habitation. Thick forest and mountainous terrain still requires traditional manual clearance and in most countries will not be cleared of mines for a long time, if ever.

Mechanized clearance methods continue to evolve with improvements to machines and techniques. Machines can be used for survey, risk assessment, and risk reduction tasks to help determine the need for more expensive manual clearance methods. Mine action programs are gradually shifting from an emphasis on total clearance in the 1990s to one of progressive prioritized risk reduction involving a series of measures including high-security fences, mechanized survey and risk reduction methods, and selective manual clearance ([48.5], part 4). Protective measures applied to agricultural machinery offer cheaper alternatives in low-AT-risk areas [48.7].

Evolution of Demining Research Priorities

Technological development in landmine clearance from within the demining community has mainly been driven by the search for improved productivity. Many of the comments in this section are based on numerous discussions with experienced demining personnel who have tried new technologies in the field. References have been cited where further detailed written information is available.

In the mid-1990s there was the expectation that, with sufficient research, advanced technology detectors could replace eddy-current metal detector technology that had been in use since the 1940s. Metal detectors also react to metal fragments in the ground. A detector that could confirm the presence of explosive, it was thought, would save having to investigate all these false alarms. The most promising line of research seemed to be data fusion: combining signals from a metal detector, ground-penetrating radar, infrared detectors, thermal neutron detectors, and even acoustic detectors. Astute observers at research conferences have pointed out that these signals were often well correlated, even in the presence of false alarms: producing a reliable detector was going to be hard work. Their forecasts turned out to be very accurate. Only one such detector is currently in operation: the handheld standoff mine detection system (HSTAMIDS) detector used by US military forces in Afghanistan employs a combination of ground-penetrating radar and eddy-current metal detection. Little information on its effectiveness has been released and no independent trials have been reported. Experienced research groups report that ground-penetrating radar requires accurate alignment of the detector with the ground surface (to

eliminate ground surface returns) and also with the target center point to enable the target to be characterized reliably. If the principal metal component of the landmine coincides with its geometric center, a common feature of minimum metal mines, the metal detector can be used for alignment. However this is not always the case and one cannot guarantee the absence of other metal fragments near the mine. Ground-penetrating radar provides confusing returns in very dry or very wet conditions and is also susceptible to false alarm indications from underground discontinuities such as stones, sticks, animal burrows etc. Publishers mostly downplay these difficulties and prefer only to report positive results, not false negatives. These issues only emerge from discussions with developers who have seriously evaluated technology in field conditions.

The major performance improvements in sensing have been obtained by compensating eddy-current metal detectors for soil magnetization, enabling them to work in a much wider range of soil conditions. Improvements in sensitivity can help with minimum metal mines but can also result in a large number of false alarms from smaller metal fragments. Metal detector arrays have been fitted to vehicles to speed up clearance of paved areas and roads [48.8].

By the late 1990s slow progress with sensors had become more apparent and research priorities after 2000 gradually turned to mine detection dogs and large demining machines.

The Afghanistan Mine Action Center (MACA) started using mine detection dogs around 1993 but it was not until 1998 that this program was running effectively. There were several difficulties. The first challenge was that close association between humans and dogs was socially unacceptable in Afghanistan. The second challenge was to devise ways to use dogs and manual mine clearance in an effective combination providing reliable clearance with high productivity. This was much the greater challenge, but by 1998 the cost of clearance using dogs was around one-third the cost of manual clearance. It was then that the problems started to appear: the occasional missed mine that could not be explained by lack of organization or failure to follow procedures. At the same time, carefully controlled trials of mine detection dogs in Bosnia had returned highly variable results. On several occasions dogs had walked past blocks of trinitrotoluene (TNT) lying almost visible in the ground. Yet, at the same time, a number of commercial demining agencies were routinely declaring land free of mines using similar dogs. In late 1999 the Bosnian Mine Action Center ran a carefully controlled test in which around 80% of

the dogs failed to achieve the required performance standard. The results were hotly contested at the time and the international community organized a systematic trial of mine detection dogs through the Geneva International Center for Humanitarian Demining (GICHD).

By 2001 it was apparent that there had been little scientific research on the fundamental physiological mechanisms that enable dogs to locate sources of explosive vapor. Dogs had been able to find mines using explosives (such as high melting point explosives (HMX)) with vapor pressure far below measurable detection thresholds. The mechanism by which TNT vapor and its breakdown products reach the ground surface was the subject of considerable scientific debate. By 2003 a systematic trial in Afghanistan, scientific studies at SANDIA Laboratories in the USA and in Scandinavia, explosive trace detection studies with dogs at Auburn University, and several other investigations provided some insight into this problem for the first time [48.9]. However, the precise physiological mechanisms for canine explosive detection remain unclear, especially for lower-vapor-pressure explosives. We do not know for sure whether dogs are reacting to vapor, minute particles of explosive suspended in the air, biochemical breakdown products, or a combination.

In 2003 a US company, NOMADICS, demonstrated the FIDO detector, the first that could reliably measure the presence of TNT vapor with greater sensitivity than a highly trained dog. However field trials showed that TNT vapor could be detected everywhere in a mine contaminated area. An explosive vapor sensor was just the beginning of the story and warns of a complex task ahead.

By 2004 the international community realized that the early confidence in a breakthrough resulting from advanced sensor technology, demining machinery, and mine detection dogs had been misplaced. GICHD commissioned the first serious study of manual demining to see whether productivity improvements could be made. A systematic series of trials were conducted in Africa to determine the effectiveness of several innovations such as magnets and rakes. The final report, issued in 2005, revealed that greatly improved productivity was possible but would depend more on improving contracting arrangements, management, and training than technology.

The New York attacks in September 2001 have fundamentally changed research priorities. Removing unexploded ordnance, particularly cluster bombs, became the top priority for the next 12 months in Afghanistan (Fig. 48.6). Since then the deteriorating



Fig. 48.6 Unexploded BLU-97 cluster bombs in Afghanistan early in 2002. Two of the small yellow canisters with parachutes still attached lie visible in the foreground. Others lie in the houses in the distance. Some may lie up to 40 cm below the ground surface. Some can detonate when a metal detector or mobile phone is used nearby. These devices have a kill radius of 200 m and can sometimes be set off by a strong gust of wind. They readily attract the curiosity of children (photo G. Zahachewsky and N. Spencer)

security and political situation in Afghanistan has focused mine clearance agencies more on maintaining security for their own workforce than trying to improve productivity and safety.

Resistance to the US and international occupation of Iraq and the easy availability of explosives both from former Iraqi armed forces and unexploded ordnance from US military operations led to the proliferation of improvised explosive devices (IEDs) to attack organized military forces and police. These have now become the main threat and the focus for much of the funding, and operational and research expertise formerly available to support mine clearance operations. This development has also placed ordnance disposal teams at the front line for the first time, rather than working in well-protected secure areas. Iraqi insurgent groups attack ordnance disposal teams both because they are attempting to disarm some of the insurgents' most effective weapons and also because they remove the main sources of explosives available to insurgent groups.

Improvised explosive devices, when detected, are often investigated and neutralized using remotely operated robots. While there are nondestructive methods to neutralize IEDs, the fastest method usually involves placing a small demolition charge on the device. Operational details remain confidential to reduce the risk that IEDs will be modified to defeat current neutralization methods.

Paradoxically it is this development that has enabled robotics to make a greater contribution to



Fig. 48.7 Bozena teleoperated demining vehicle (Way Industry, Slovakia)

the problem by contributing improvements in remote manipulation technology. These improvements come more in the form of low-cost commercial off-the-shelf components (mobile platform, motors, TV cameras etc.) than from fundamental research advances. Improvements are still being made: improved remote manipulation, blast survivability, operator interface improvements, and mobility improvements have all contributed significantly to performance and reduced operating costs.

Advances in Demining Robotics Research

In a brief survey it is not possible to mention every contribution. We have attempted to provide a sample of research reports that illustrate the main achievements and we present brief technical discussions on mobility and manipulation dexterity.



Fig. 48.8 Gryphon experimental robot on trial in Croatia (Photo S. Hirose) [48.10]

Teleoperation remains the only robotics technology that has been used in practical application in field conditions [48.11, 12]. Robotics research has not yet been able to make a significant contribution to mine and unexploded clearance work. However, the problems posed by landmine clearance have stimulated new research results that could have other applications.

Robotics researchers started their efforts in the early 1990s, for example, Stephan Havlik and *James Trevelyan* independently proposed suspended cable robots to work in minefields [48.13]. However, both have later argued in favor of alternative solutions [48.11, 14].

Nicoud [48.15] wrote one of the first surveys exploring the possibility of using robotics technology for landmine clearance. Developments in robotics research since the mid 1990s have been motivated both by a genuine desire to help combat a serious humanitarian issue and also by a desire to find a justification for more fundamental research. However, most researchers still have not learned lessons from the field such as the need to remove vegetation and the variety of situations in which deminers find themselves [48.16].

One of the most prolific research areas justified in part by humanitarian demining problems has been path planning for autonomous agents [48.17]. Probabilistic approaches were explored by [48.18]. Some researchers have proposed multiple robot solutions, even *swarms* of robots [48.19] and so-called *immune systems* [48.20]. Autonomous search and mapping algorithms have also been explored [48.21], including even three-dimensional search techniques for locating underground chemical sources [48.22].

Landmine clearance has also stimulated developments in autonomous robot vehicles, with many examples in the literature [48.23]. Tracked vehicles have been proposed for working in rubble and built environments [48.24]. Walking vehicles have been proposed, particularly for difficult terrain in countries like Bosnia and Afghanistan [48.25] even to the extent of examining how damaged robots with missing legs could extricate themselves from a mined area [48.26].

The desire to keep human operators away from the risk of handling unexploded ordnance has stimulated research on artificial hands and telemanipulation [48.27]. Purely mechanical devices have also been explored [48.28]. Several researchers have provided detailed results of tests with manipulators mounted on autonomous vehicles [48.10].

Robotics solutions have also been proposed partly to overcome the limitations of handheld sensors such as ground-penetrating radar. A robot manipulator can

control the motion of the sensor so much more precisely, opening the possibility of synthetic aperture techniques for both metal detection and radar [48.29].

Military research agencies in the United States, Australia, Britain, and Canada devoted large research budgets to the problem of road clearance after military casualties in Somalia and Bosnia. Insurgent forces had demonstrated that they could bring organized military forces to a complete standstill overnight simply by laying antivehicle mines in a few road potholes. With most roads in poor condition there was no easy way to detect that the mines had been laid. Insurgents would typically use mines to stop leading vehicles in a convoy in order to increase the effectiveness of an ambush.

Most research teams proposed one or more vehicles carrying multiple sensors including ground-penetrating radar, metal detector arrays, passive and active infrared, and even some acoustic arrays. Some sensor arrangements were designed to look forward sufficiently far to allow the carrier vehicle to stop before reaching a mine. Others were to be carried by lightweight remotely controlled lead vehicles. Military planners calculated that search speeds of approximately 30 km per hour would be required to be able to check roads daily in time for supply convoys to use the roads in daylight hours. Many different arrangements have been reported [48.30]. Other teams proposed teleoperated devices for landmine and explosive ordnance detection and neutralization but few anticipated the requirement to operate under armed attack. Typical requirements envisaged secure rear area mine clearance in combat situations and peace-keeping on roads and tracks [48.31] or force protection roles [48.32]. Most of these efforts were initiated in the late 1990s but by 2005 it had become apparent to military planners that vehicle protection rather than mine detection was a more practical solution. Much of today's vehicle protection technology originated in southern Africa with further development in Australia, the UK, and other countries, stimulated in part by South African expatriates.

One way to reduce the distance between researchers and field problems, at least in terms of geographic distance, has been to promote research in landmine-affected countries such as Sri Lanka and Colombia [48.33]. However this is not easy. Most countries affected by landmines have been disrupted by social conflict and destabilization that led to the military conflicts in which landmines were used. This makes it difficult for local people to create sufficient economic and physical security for researchers to pursue their work.

Future Prospects For Robotic Demining

What are the challenges for robotics researchers working on landmine clearance and other hazardous applications in the future?

We need further advances in mechatronics design, sensing, and accurate understanding of the problems to be solved using robots.

The best starting point for research is to witness people undertaking hazardous work in several different situations. Nuclear accidents, mine disasters, and burning buildings are usually off-limits to researchers. However, mine clearance operations are readily accessible in many countries. It is unfortunate that many researchers think a visit would be far too hazardous and, as a direct result, have failed to appreciate the practical difficulties involved. Photographs taken at mine clearance operations are available to provide researchers with a web site for reference purposes, partly in answer to this need to understand the practical realities [48.34].

One of the main motivations for robotics researchers has been the perception that mine clearance is a hazardous occupation and that it would be more preferable for robots to be exposed to minefield risks than human beings [48.35]. While mine clearance is certainly a hazardous occupation it is not necessarily dangerous. Accident records show that mine clearance in Afghanistan in 1998 resulted in about half the rate of injury of the United States forestry industry and about one third the rate of injury for the United States building construction industry per 100 000 working hours. Mine clearance agencies use advanced techniques to improve safety when possible [48.36]. In terms of deaths, demining is considerably less hazardous than mining, construction of building foundations, and especially offshore drilling rigs ([48.37], pp. 11–14).

Another motivation for research is to reduce deaths and injuries among local people who have to live with the daily threat of landmines and unexploded ordnance. Again, there are misperceptions of risk. The incidence of death and injury from mine explosions is often very small compared with disease, for example. The main priorities for local people tend to be improvements for water and food supplies, education, sanitation and physical security: landmine clearance is usually a much lower priority and it is often hard to justify significant local resources.

It is also important that robotics researchers intending to contribute to the solution of this problem understand the relatively small size of humanitarian demining operations, which have been funded from a combined international humanitarian aid budget of

approximately US\$400 million. These programs spend an estimated \$20 million annually on all equipment needs. The market for specialized humanitarian demining detectors is therefore very small and manufacturers cannot afford research and development specifically to support humanitarian demining solutions [48.38]. Adapting technology developed for other purposes, such as military equipment or civil engineering construction machinery, is more likely to be feasible.

The last 10 years has seen significant improvement in mine clearance techniques but progress is still slow and robotics may well provide the final solution in the long term. There is plenty of time to develop robotic techniques that could ultimately provide the only cost-effective method for removing this menace.

48.2.2 Hazardous Materials Handling and Operations

The oldest application of robotics-related technology to hazardous environments is various aspects of remote nuclear operations dating back to the beginning of serious work on atomic physics in the early 1940s. This section discusses hazardous materials and operations in the context of nuclear applications and some extrapolations to other domains. These applications run the spectrum from low- to high-fidelity manipulation and multiple mobility modes.

Many variations of remote handling systems have been in use since man has attempted to cope with hazardous environments. In the 1940s, research in atomic physics led to a new era in remote handling as scientists sought to explore the nature of materials involving ionizing radiation. As experiments became more complex, mechanical manipulator systems were created which allowed operators to perform increasingly complex tasks safely behind thick biological shielding. These mechanical systems then evolved into electrical systems that allowed larger work volumes to be considered. Incredible engineering achievement occurred in a 15 year period within the Remote Control Division of the Argonne National Laboratory. Even though this era represented tremendous technical achievement, it went further in illustrating the intrinsic complexity of remote operations. The equivalent work performance achieved with sophisticated teleoperated remote systems is poor in comparison to what human workers can achieve with direct contact operations and common tools. Typically, this form of teleoperation (i.e., manual control over a physical distance or barrier) is ten to hundreds of times slower than conventional contact operations. Remote

operations are extremely expensive and time consuming and have been the continual target of engineering improvements over the years.

Many research and development efforts have focused on different avenues for improving the work efficiency of teleoperated remote operations. These efforts have included the development of better manipulators, control stations, control algorithms, etc., all intended to enhance reliability and maintainability. In the late 1960s and early 1970s, as digital electronics became more cost effective, interest began to emerge in the integration of automation with teleoperation as a scheme to effectively increase remote operations work efficiency. It was around this time that industrial robot concepts were also introduced. Combining selective automation of specific subtasks with traditional teleoperation offers the potential to reduce labor requirements and to improve the quality of repetitive task executions. This integration of automation with teleoperation became the foundation of what is now termed *telerobotics*. From the 1970s until today, telerobotics has been an active area of research and development in many different domains that include nuclear, space, and military applications. Unlike manufacturing automation, remote operations in hazardous and unstructured work task environments necessitate human-in-the-loop control, or teleoperation, as a backstop to assure safe operations. Teleoperation related to hazardous materials and operations, in the most general sense, involve mobility and the use of manipulators to handle objects and tools to accomplish useful work in predominantly unstructured and uncertain environments.

Over the years, there have been numerous papers, books, and reports written concerning technical challenges, issues, and solutions. Readers will find *Vertut* [48.39] and *Slutski* [48.40] provide comprehensive and general information about manipulators and systems. The performance and design of effective teleoperated systems are strong functions of human factors and *Kraiss* [48.41] and *Johnsen and Corliss* [48.42] provide good discussions of key principles. Finally, control system hierarchies and structures are explained in depth by *Sheridan* [48.43].

Past Perspectives

Initially, science and experiments involving dangerous radiation levels were accomplished through the innovative use of shielding walls, long-handled tools, and mirrors. As the tasks became more difficult, mechanical arms that could do a better job in emulating human motions were pursued. These ideas represented schemes

whereby human sensing and handling capabilities could be more completely *projected* into the remote work environment [48.44]. One of the first challenges that the Argonne group tackled was the development of the mechanical master–slave manipulator. The basic concept was to create a mechanism that would have a master controller side where an operator could provide position and orientation commands to a slave-side mechanism/linkage system that would *replicate* motions and forces in the remote work area. It was felt that force reflection to and from the master and slave systems was essential for the operator’s sensory awareness of the task execution. Experimentation has repeatedly verified the significance of both kinesthetic and tactile feedback in performing more complicated tasks [48.45].

Today, master–slave manipulator (MSMs) are used around the world in nuclear, biological, and other types of hazardous remote experimentation and operations. The remote work efficiency of a dual-arm MSM with shielded-window viewing is around 5–10 times slower than equivalent contact operations. Because the master and slave sections of MSM’s are mechanically coupled through the metal-tape drive transmission, the physical separation that can exist between the safe operating area and the hazardous remote work area is limited to a maximum of approximately 10 m. Because of this characteristic and their kinematics, MSMs are restrictive in many applications and often have constrained the physical design of remote cells. The Argonne group and others recognized that it would be much better to have the equivalent of a fly-by-wire MSM in which the physical separation of the master and slave could be larger. This need led to the development of electrical master–slave manipulators (EMSs) that are commonly called electrical servomanipulators [48.46, 47]. Research and Development on the EMS began in the late 1940s and continued into the early 1950s.

The Argonne group was at that time limited by the available electrical control technology. Nonetheless, they made noteworthy progress toward integrated systems, as depicted in Fig. 48.9. The prototype system shown is a dual-arm anthropomorphic system with head-aiming remote television viewing and bilateral force reflection. After the Argonne Remote Control Division was disbanded, limited research and development occurred until the 1970s when commercial nuclear power growth was driving a number of research programs in the US, West Germany, France, and Japan. During this time, the programs in the US and France were focused on electrical servomanipulator systems which incorporated emerging microprocessor technology. The Central

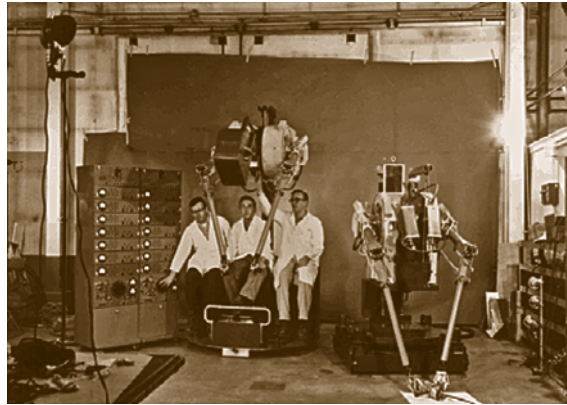


Fig. 48.9 An integrated electrical master–slave manipulator system (courtesy of Oak Ridge National Laboratory)

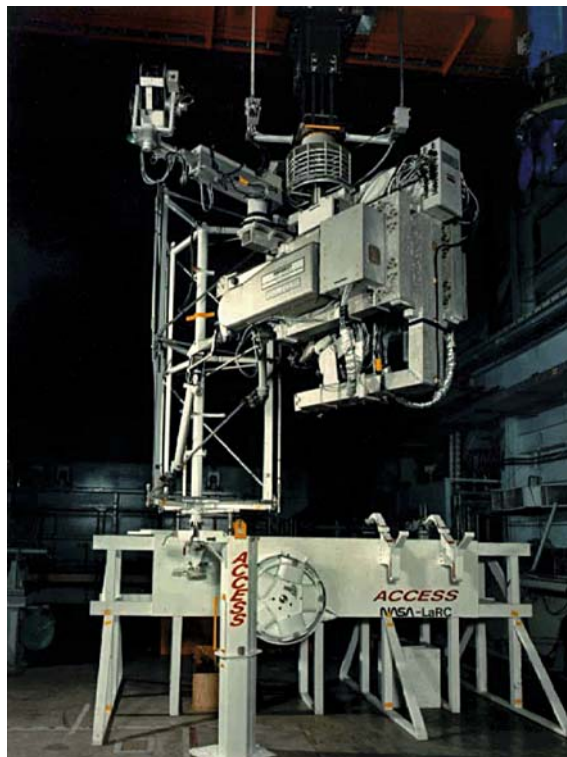


Fig. 48.10 CRL Model M2 manipulator system performing space truss assembly (courtesy of Oak Ridge National Laboratory)

Research Laboratories model M2 system, shown in Fig. 48.10, was jointly developed with the Oak Ridge National Laboratory and was the first force reflecting servomanipulator system to use distributed digital



Fig. 48.11 Advanced servomanipulator system (courtesy of Oak Ridge National Laboratory)

electronics to implement position–position force reflection with multiplexed serial communications between the master and slave. The model M2 system was used over the years to explore remote operations for military, space, and nuclear applications.

The development of the advanced servomanipulator (ASM) followed the M2 in an effort to improve the remote maintainability of the manipulators themselves. This work was one of the earliest modular robotics efforts. The motivation for this work was to reduce maintenance technician radiation exposure and to increase the overall availability of the remote maintenance system. The ASM was designed from the beginning to provide a foundation for telerobotics in addition to effective teleoperation [48,48] (Fig. 48.11).

At the time of the M2 and ASM developments, Jean Vertut and his colleagues in the Commission de Energie Atomique (CEA) focused their research on the development of telerobotic functionality for their MA-23 electrical servomanipulator systems. This

research appears to include some of the earliest experimental demonstrations of telerobotic functions [48,39]. They called their concept computer-assisted teleoperation and it included both operator assists and robotic teach/playback functions. Operator assists included software jigs and fixtures designed for the improvement of the remote operation of tools such as saws, drills, etc.

In the 1980s and 1990s as nuclear power activities began to decline, nuclear remote operations technology was migrated into other areas such as space and the military. Nuclear remote operations experience with teleoperation influenced the Space Shuttle remote manipulator system and the short-lived Flight Telerobotic Servicer program.

In the mid 1990s, substantial interest developed in the application of remote handling systems to deal with problems in hazardous site/facility remediation, such as those associated with defunct nuclear facilities in the US and in Eastern Europe. The intrinsic complexity of remote clean up operations continues to drive research and development in all aspects of teleoperated and telerobotic systems. Challenges and accomplishments in this application area are discussed in the next section.

Hazardous Site Clean Up Applications

Robotic and remote systems have been used in assessing the status of sites contaminated with hazardous materials. These surveys are essential in planning and executing subsequent clean up operations. Such systems have been used most extensively in nuclear waste sites around the world because remote techniques are more common in many nuclear operations. There has been some use of robotic survey systems in chemical and biological hazard situations as well.

Survey Systems. Numerous robotic survey systems have been developed and used. The basic idea is to integrate a sensor suite, appropriate for the contaminants of interest, with a suitable mobile platform that incorporates remote and/or autonomous driving functions and requisite navigation and control functions. The desired output of the survey process is a *precise* map of contaminant locations and concentrations. Such systems have been developed for both outside and inside operations.

It was common practice in many industries for a number of decades to bury hazardous wastes in earthen trenches in isolated burial sites. Usually, useful records of what materials were buried at what locations either did not exist or were not accurate. In fact, the general conditions of such buried waste sites are often unknown to the extent that human entrance is not allowed. As a result, the first step toward remediation is to quantitatively

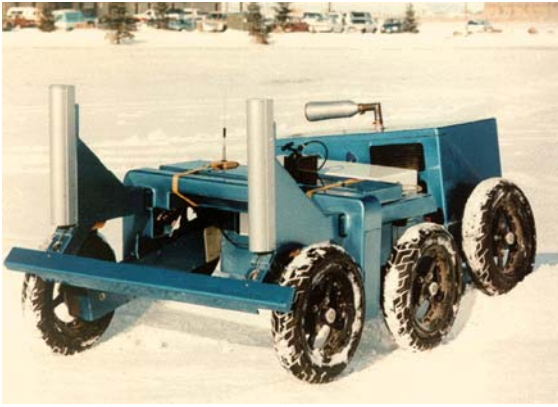


Fig. 48.12 Prototype buried waste survey robot (courtesy of Los Alamos National Laboratory)

assess the physical and hazard conditions of the site. The mobile robot shown in Fig. 48.12 is an example of a survey system used to evaluate nuclear waste buried waste sites. The robot's location is monitored with Global Positioning System (GPS). The suite of sensors includes eddy-current probes and ground-penetrating radar that reveal density contours, radiation detectors, and gas emissions monitors. This system was an initial prototype that could be operated from a remote driving station or operated in autonomous programmed trajectory mode. The system was controlled using a radio communication link. Its unique feature is that it was designed with minimal use of ferromagnetic materials to minimize interference with the magnetic subsurface sensors.

The mobile characterization system shown in Fig. 48.13 is a similar concept for radiation survey of flooded areas. This particular system uses a Cybermotion commercial mobile platform, triangulation of optical



Fig. 48.13 Mobile characterization system (courtesy of Oak Ridge National Laboratory)

fiducial markers for localization, and operates primarily in a full autonomous mode. The objective in this application is to reduce labor costs and eliminate error-prone tedious human operations.

Excavation Systems. The actual remediation of buried wastes ultimately leads to digging and object-handling type operations. Various types of systems involving integrated mobility and manipulation-like functions have been developed. One of the most popular and low-cost approaches has been to retrofit conventional excavation equipment with sensor and actuators that allow remote and robotics operations. Such systems have been used in explosive ordnance disposal also. The system shown in Fig. 48.14 is called the teleoperated small emplacement excavator (TSEE) and is a prototype remotely operable backhoe for such applications. The unit included a multicamera remote viewing system that is the primary basis for teleoperation. Radio and tether communica-



Fig. 48.14 Remotely operated excavator (courtesy of Oak Ridge National Laboratory)



tions connections are used to allow the human operators to be displaced up to several miles from the hazardous operations.

Deactivation and Decommissioning Systems. In recent years, one of the most complex forms of hazardous operations is the deactivation and decommissioning (D&D) of defunct facilities where nuclear radiation or toxicity hazards preclude human presence. D&D can be thought of as remote demolition for the most part. Some operations are crude such as knocking down building structures and debris removal. Other operations may involve careful disassembly of equipment and devices, size reduction and packaging of handling/storage. These operations are essentially the inverse of remote maintenance and require the dexterous use of tools and handling of objects. Tools include saws, hydraulic shears, impact wrenches, and the like. Planning and situational awareness are very important in demolition-type operations because the physical layout and stability of the basic environment is changing during the process. Effective remote viewing (adjustable multiview capabilities) and acoustic sensing are essential. Audio feedback from the remote environment provides the human operator with the ability to monitor the normalcy of tooling operations such as sawing. D&D applications require manipulators that are both dexterous and massive. Experience has shown that payload capacities on the order of 100 kg are needed to deal with debris handling, typically with manipulators of the type shown in Fig. 48.15. Figure 48.16 is a frame view from one of the remote viewing cameras used to operate a similar system that was used to D&D an old experimental nuclear reactor. A manipulator is being used to teleoperate a conventional circular saw to segment a large-diameter aluminum tank. This operator view of the remote environment provides a realistic perception of the task environment lighting conditions and complexity.

In recent years, numerous remote manipulation systems for nuclear applications have been developed



Fig. 48.15 Dual-arm D&D manipulator system (courtesy of Oak Ridge National Laboratory)

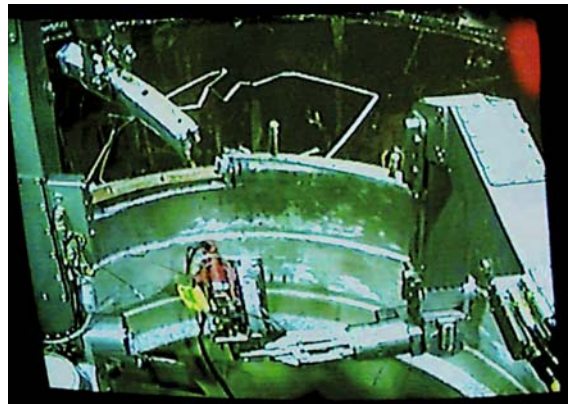


Fig. 48.16 Operator's view of a remote sawing task (courtesy of Oak Ridge National Laboratory)

around industrial robots. These systems are augmented with sensors and master controller arrangements to facilitate teleoperation. This approach offers significant cost savings and the level of teleoperation performance achievable in complex task environments has steadily increased [48.49–53].

48.3 Enabling Technologies

There are a number of technology issues that are key to the design and implementation of mobile robots that can operate effectively in different types of hazardous environments. A few of them are touched upon here to give the reader additional insight into the foundational aspects of this class of robots.

48.3.1 Mobility Issues

Hazardous application such as fire fighting, EOD, and demining naturally include uneven ground in indoor and outdoor environments where a normal wheel-type robot cannot easily operate. It is critical to have a mobile

mechanism with a good negotiability in uneven ground conditions. Most mobile mechanisms are classified into wheel type [48.54–57], track type [48.58–63], and leg type [48.64]. For negotiability in uneven ground, the wheel and track types need additional linkages with ground adaptation. There are two kinds of this adaptation: active and passive adaptation. Active adaptation uses an additional actuator to alter the linkage's motion while in passive ones the linkage motion is controlled by the ground conditions and gravity effects.

Stability Conditions

Stability should be considered in the design process to prevent rollover on uneven ground including stairways, steps, and natural terrain. Stability can be investigated in terms of three parameters: *center of mass*, *supporting area*, and *stability margin*. Stability requires that the center of mass remains inside of the supporting area, as shown in Fig. 48.17. The supporting area is a polygon built by the edges connecting each projection point on the horizontal plane. A conventional vehicle has limitations in rugged terrain due to a fixed center of mass with respect to the body coordinates. The center of mass greatly affects the stability margin (the minimum length between the center of mass and edges of a supporting area). If the center of mass is located out of the supporting area, rollover occurs. The stability margin of a conventional vehicle is mainly determined by the inclination of landform. Many vehicles are designed to have a low center of mass, thereby obtaining a large stability margin on a slope. Furthermore, a vehicle is often designed to have multiple bodies to overcome this limitation. For a multiple-link mechanism, the center of mass varies and the relative motion of links when traveling over the landform also alters the supporting area.

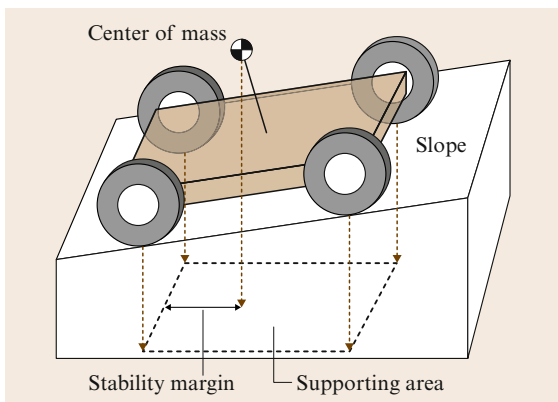


Fig. 48.17 Stability condition

Active Adaptation Mechanisms

An active adaptation mechanism uses an additional actuator to generate motion of the adaptation linkage on rugged ground. Although additional actuators and linkages are required, the adaptation mechanism can overcome variously shaped rugged landforms.

It is known that a vehicle can negotiate an obstacle which is smaller than radius of a wheel. One may think that a vehicle which has large wheels can pass over rough terrain like a monster truck. However, most applications require a small vehicle to drive between obstacles.

A deformable wheel or its equivalent mechanism can be adopted: insufficiently inflated wheels, as shown in Fig. 48.18a, adapt to irregular terrain by deformation of wheels. However, their size should be big enough to negotiate rough terrain, so this concept still has size limitations. Their equivalent mechanisms are relatively compact and yet they need sophisticated articulation mechanisms, as shown in Fig. 48.18b. Two wheels are attached to a link that rotates about a main body. Therefore, revolution and rotation of wheels are made as shown in Fig. 48.18b. If rotation is provided by actuators at the main body (i.e., an active mechanism), the vehicle has an equivalent wheel whose radius is equal to that of the rotation. The vehicle in Fig. 48.18b can pass a sill via carriage rotation, although the radius of a wheel is smaller than the height of the sill.

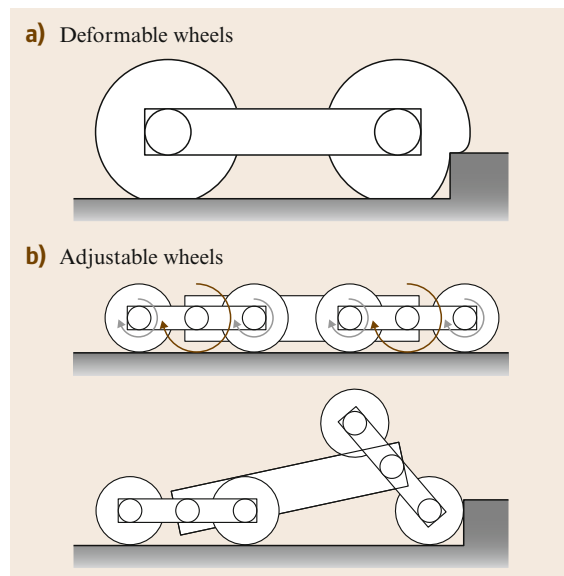


Fig. 48.18a,b Wheel configurations (a) deformable wheels, (b) adjustable wheels

a) Packbot [65] **b)** Andros Mark V-A1 [61]

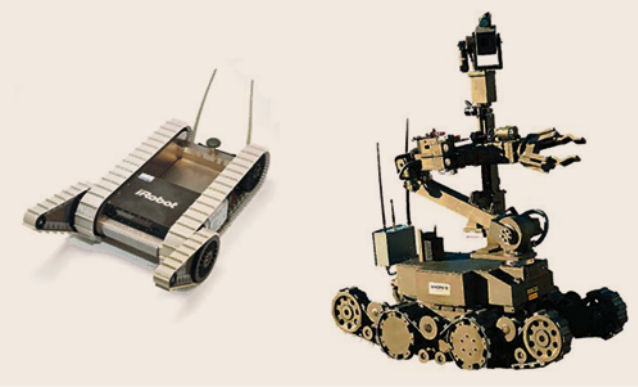


Fig. 48.19a,b Mobile robots with active adaptation mechanisms
(a) Packbot [48.65], **(b)** Andros Mark V-A1 [48.61]

Many track-type mechanisms have multiple linkage structures with active adaptation [48.60–63, 65]. When traveling over uneven ground, these mechanisms can negotiate rugged ground by changing the configuration of multiple track mechanisms, for example, typical multiple-tracked robots with active adaptation such as *Andros* [48.61] and *Packbot* [48.65], as shown in Fig. 48.19a and b, commonly use a small active additional track called a flipper. By using the flipper when spinning on the ground the robot reduces undesired friction torque between the ground and the track by lifting up the flippers. When climbing up stairs, touching the flippers down to the ground can increase the stability by enlarging the supporting area. Snake-like robots use active interconnecting joints [48.62] and have the potential for mobility and manipulation.

Passive Mechanisms

Passive adaptation mechanisms do not use any additional actuators; rather they simply utilize the gravity effect to generate adaptation to irregular ground. This approach is less adaptive but enables the operator to drive the robot easily because he (or she) does not have to take care of controlling the adaptation mechanism. Passive mechanisms usually include various kinds of passive linkages activated by irregular ground conditions.

For the wheeled vehicle shown in Fig. 48.18b, a passive mechanism can be designed in which no additional actuators are needed, for example, when a wheel meets a sill whose height is greater than the radius of a wheel, it will become stuck. Then high torque is exerted at the wheel since the stall torque is usually greater than the normal rotation torque of the wheel. In this case,

a) Deformable wheels



b) Adjustable wheels



Fig. 48.20a,b Mobile platforms with passive adaptation: **(a)** Soleo-Shrimp [48.57] and **(b)** Robhaz-dt3

a closed linkage without any actuator except the driving wheels can generate a passive motion to climb over the sill [48.55], for example, a smart mobile robot with passive adaptation called *Soleo-Shrimp* (depicted in Fig. 48.20a) can climb up an obstacle by lowering the instant center of rotation of the passive four-bar linkage when coming into contact with the vertical surface of the obstacle [48.57].

Similarly, multiple bodies with tracked and chained passive joints can also adapt to irregular landforms. The track type is inherently insensitive to the unevenness of ground. Thus passively chained multiple-tracked bodies can provide greater reliability and better capabilities in ground adaptation. For example, a mobile robot with a passively chained double-tracked mechanism called *Robhaz* (shown in Fig. 48.20b) is a practical design with

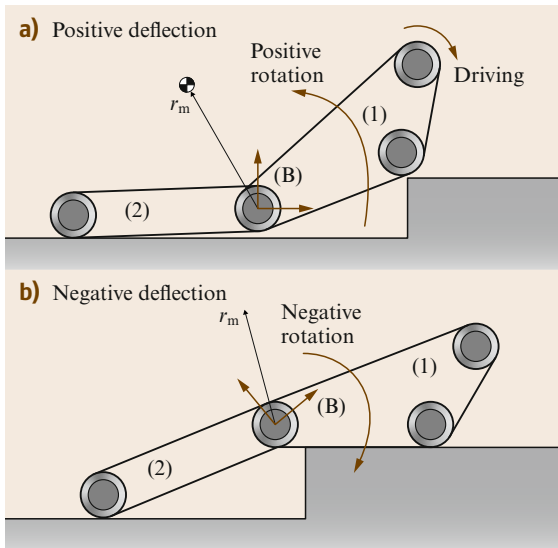


Fig. 48.21a,b Passive adaptation of double-tracked mechanism: (a) positive deflection and (b) negative deflection

simply adapted multiple tracked bodies with passive joints [48.63]. As depicted in Fig. 48.20, two tracks of the Robhaz rotate positively or and negatively according to the contact situation between the track and the stair surface. Based on this behavior, in terms of the lower center of mass (Fig. 48.21a) and the supporting area (Fig. 48.21b) better stability in stair climbing than that of a single-tracked vehicle can be achieved.

48.3.2 Manipulator Design and Control for Hazardous Object Handling

Manipulation is essential for EOD missions. Generally, EOD missions have two phases: (1) approach, and (2) object manipulation to achieve ordnance neutralization. The second phase is generally controlled by teleoperation. Both the human's careful control and innate intelligence, plus appropriate manipulator design and control, are needed for such dangerous tasks.

Design Requirements

Unlike industrial manipulators, the design of manipulators for many robots in hazardous environments does not require very fast movement. Instead, high payload, lightweight, and compactness are more important. Therefore, many manipulator designers locate actuators and reducers, which typically have high weight and large volume, on the base or lower links. This reduces the weight of the moving parts of the manipulator



Fig. 48.22a,b Hazardous application robots with manipulator: (a) Packbot with manipulator and (b) Robhaz-dt2 with manipulator

and offers better stability and dynamic control properties. Consequently, they have a thin profile, as shown in Fig. 48.22a,b. This characteristic is usually associated with explosive ordnance disposal-type robots that require access to congested areas and normal human living spaces. Mobile robots used in other hazardous environments can be much larger, depending on the specific applications. Refer to Sect. 51.2.2 for examples.

48.3.3 Control for Hazardous Tasks

In most cases, the operator does not have much information about the environment around the robot and the manipulator. He may have very limited vision, and unexpected collisions between the manipulator and obstacles can easily happen, even if he (or she) tries to be very careful. The robot should be able to handle these risks by appropriate design of the manipulator and good con-

trol algorithms. There are several ways to solve these problems.

Proximity sensors that detect objects around the robot and/or mounted on the manipulator can be used to avoid a collision before it occurs [48.66]. However using many proximity sensors necessitates large wiring bundles and complex manipulator design. Force control is also a good way to handle collisions. When a collision occurs, the manipulator can move away from the collision point by using force/torque sensor data. Often a six-axis force sensor attached at the end-effector of the manipulator is used for the force control [48.67]. Many researchers have developed stiffness/compliance/impedance control methods for a manipulator with a force sensor. However, this method can only handle a limited collision zone – only the end-effector or the manipulator *hand* – because the sensor cannot sense contact with intermediate links, while a collision could occur at any point on the manipulator. For this reason, some manipulators use joint torque sensors at all of their joints [48.68]. When using a joint torque sensor, contact at any point on the manipulator can be sensed, however, gravity compensation and errors in the transformation from the joints to the Cartesian space introduce further difficulties.

The stability of telerobotic systems, particularly when force feedback or reflection is involved, remains a critical issue. Systems involving significant data communication time delays are even more difficult. National

Aeronautics and Space Administration (NASA) groups have studied stability and bandwidth issues numerous times over the years; one of the more recent studies is provided by Uebel et al. [48.69]. Energy methods have been applied to this problem, resulting in new ideas entitled *passivity* control that show promise for enhanced stability and performance [48.4, 70].

Master Controllers for Teleoperated Manipulation

In teleoperation, the robot is designed to be a faithful slave to deal with a dangerous task while the operator uses a control interface to direct the slave from a safe location. The user interface usually provides a means for

- sending position commands to the manipulator
- providing contact force feedback or force reflection to the user

Ideally, the design goal of a user interface is to make it *transparent*, so the operator feels as if he or she is directly manipulating the object handled by the slave manipulator. To achieve this transparency, there are several design issues

- Simplicity:
 - All indicators are unified as one scene.
 - All input button and joystick are integrated into one haptic device.
- Intuitiveness:
 - High-level command by speech recognition.
 - Human-friendly feedback such as graphical displays and human voice.
 - Motion command matching between the haptic device and the slave in Cartesian space.
- Portability:
 - Small/lightweight and human-friendly design for wearability
 - Tetherless operation without communication lines and additional power.

An example of wearable multimodal user interface is provided by [48.71], as shown in Fig. 48.23. The operator wears a head-mounted display (HMD), head tracker, and headset to interact with the slave. A six-degree-of-freedom haptic master is attached on his waist together with the standalone controller, and the operator grips its handle to telemanipulate in Cartesian space. All the control hardware including the batteries are packed into one backpack, so that the user can walk around to gain a better view during teleoperation. It includes radiofrequency (RF) and wireless local-area network (LAN) modules enabling completely

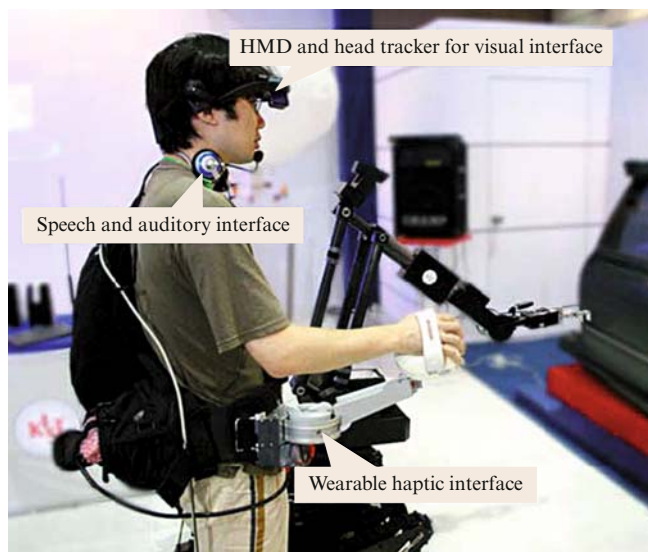


Fig. 48.23 Appearance of proposed wearable multimodal user interface

wireless communication. It is composed of major three interfaces.

Speech and Auditory Interfaces

An operator usually sends two types of commands to the robot: selection and continuous motion commands. For example, the selection between mobile and manipulation mode, the reset of the robot arm and mobile base, the on/off and reset of pan-tilt motors, speed selection for the mobile platform, selection among installed cameras are defined in the selection commands. It is convenient if the operator can issue these commands by speaking to the controller. When the operator says a word that has been predefined as a command, the speech interface can sense that word and, if the speech recognition system successfully recognizes it, the recognized command pops up on the HMD for confirmation. Finally, the operator decides whether to execute or cancel the command with a confirmation button on the haptic master.

The auditory interface synthesizes the human voice using a speech synthesis engine. It can warn of the approach of an obstacle by sound, or reveal the distance and direction to a laser-designated object nearby.

Wearable Haptic Interfaces

A wearable haptic device is shown in Fig. 48.23. The base linkage is designed as a serial RRP mechanism to measure a translation, and an RRR z - y - z rotation mechanism is attached at the end of the base linkage.

Figure 48.24 shows a wearable haptic master device for teleoperation of mobility and manipulation systems. It has six degrees of freedom for motion input and three degrees of freedom for force feedback.

To achieve a compact design and reduce its weight, a tendon-driven mechanism is designed into each joint.

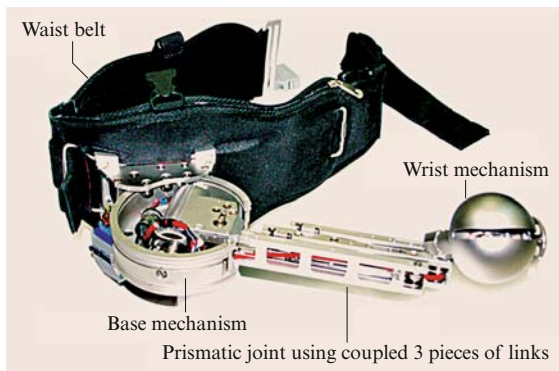


Fig. 48.24 Picture of wearable haptic device

Due to weight constraints, only three actuators are installed for force feedback, and each actuator is specially designed to fit the joint. Because a passive actuator is better than an active actuator with respect to power density (power per unit volume or weight), small magnetorheological (MR) brakes have been developed. They are installed at each joint of the base linkage for force feedback. Also a compact brake drive with current feedback capability has been designed, which reduces the response time of the MR brake. The controller is packed into a bag that is attached to the back side of the waist. Because it includes brake driver module, satellite controller, wireless LAN module, and battery, it can operate without cable connections.

Visual Interfaces

The visual interface integrates the robot's view, the status of sensed data, and the status of speech commands. Since the slave includes a stereoscopic camera, the user sees a three-dimensional view on the head-mounted display (HMD). The operator wears the head tracker and it generates a pan/tilt command from the two-degree-of-freedom head motion. In the integrated system, head tracker data is used to command the direction of the remote viewing camera, thus the user easily

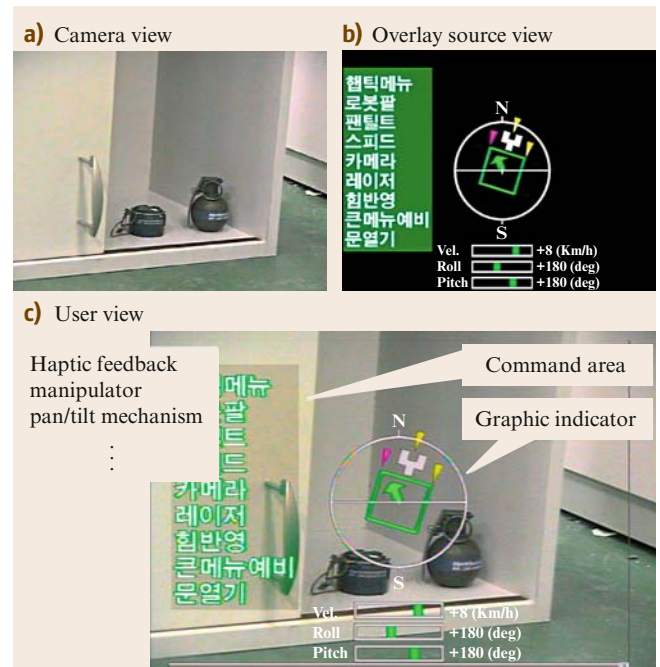


Fig. 48.25a-c Appearance of integrated visual interface: (a) camera view (b) overlay source view (c) user view

looks around the environment of the robot, as shown in Fig. 48.25.

Moreover, the head tracker is useful to indicate an object as a target. The operator moves his head and looks at a target to get information of direction and distance, then triggers a laser displacement sensor that is placed in parallel with the pan-tilt camera. The vision information is sent via a RF channel while the sensed data is fed back via an independent wireless LAN channel to reduce the data communication traffic. A source for video overlay is prepared with the reported data, as shown Fig. 48.25. The recognized speech command is highlighted on the left for confirmation. When an ultrasonic sensor detects an obstacle, it shows a round robot icon in the middle of the scene. Other useful and important information (i. e., velocity, heading direction, view direction, arm posture, etc.) are shown by bar graphs. These are overlaid on the remote video source pictures and unified into a single scene. Finally the operator sees the stereoscopic picture and the status of the robot at a glance on the immersive HMD. The overlaid view is shown in Fig. 48.25c.

48.3.4 Data Communications

Robots used in hazardous environments are almost always mobile so that they can move about an area of interest with flexibility. The combination of mobility and intrinsic operation across a physical or hazard barrier introduces unique problems with respect to bidirectional data communications (refer to Fig. 48.1). Communications from the operator location to the robot is necessary to achieve remote control or teleoperation. Communications from the robot to the operator are essential to

connect the operator’s perceptive skills into the remote environment. The transmission of the electrical and/or optical signals comprising such data communications can be quite complex due to a host of factors such as penetration of physical barriers, signal transport delays, signal attenuation, and data throughput requirements. As summarized in Table 48.1, data throughput requirements are dominated by the feedback of visual imagery or remote camera views from the mobile system. Channel capacity necessary for audio feedback and control are small in comparison. If the mobile system involves precision manipulation with force feedback, bidirectional data channels on the order of 1 Mb/s are required.

For a mobile robot with high-fidelity manipulation, multichannel remote viewing, and other sensors, on the order of hundreds of megabytes per second are required for the data communications path from the remote environment to the operator station while on the order of ten megabytes per second are needed for the bidirectional control link. Communications at these data rates are straightforward with the use of retrievable hardwired cabling or tethers. Tethered systems complicate mobility and reliability significantly. Tethers that combine signal and power conductors are difficult to pay out even with servo-controlled deployment/retrieval reels and are often snagged on obstacles such as debris. Free-space radio or microwave signal transmissions at these rates in the cluttered environments often associated with hazardous robots are much more difficult and require specially designed systems. Data throughput requirements can be reduced through data compression and reduced frame rate schemes with effective results in many cases.

Table 48.1 Example data communications requirements

Type	Single channel	Typical system
Standard black-and-white remote television: 600×400 pixels 30 frames/s 12 bit gray scale	≈ 10 Mb/s per viewing channel	30–50 Mb/s for 3–5 channels
Color remote television: red, blue, and green = 3× black and white	≈ 30 Mb/s per viewing channel	90–150 Mb/s for 3–5 channels
Control 12 bit resolution input and output 200 Hz sampling rate	≈ 4.8 kb/s per control channel	48 kb/s - ten control servo channels
Audio feedback 15 kHz signal capture 12 bits resolution	≈ 180 kb/s per audio channel	540 kb/s - three remote microphones

48.3.5 Energetics

Another key challenge in fielding mobile robots in hazardous environments is in the area of energetics including power supply, consumption, conversion, and management. The specifics of the given environment often restrict the types of power supply/conversion that are permissible, for example, the use of combustible fuels and internal combustion engines is seldom allowed within nuclear facilities but are entirely acceptable in demining and most outdoors operations. The most common power systems used today are electrochemical batteries and electric motor drives. The power and energy density map [48.72] given in Fig. 48.26 shows the fundamental situation for mobile robots used in hazardous environments. For perspective, a small passenger automobile is estimated to need a minimum of 200 Wh/km to meet road demands, which in turn translates into 500 Wh/kg. This is why hybrid electric vehicles that combine spark ignition engines with battery-powered electric motors are becoming popular. A large mobile robot with dual manipulators operating in the payload range of hundreds of kg would have power requirements similar to these and would be well above the 100–200 Wh/kg range that batteries can provide. A force-reflecting six-degree-of-freedom robot manipulator with a 20 kg payload will have a peak power consumption on the order of 10 kW. If a particular application domain is amenable to the use of spark ignition

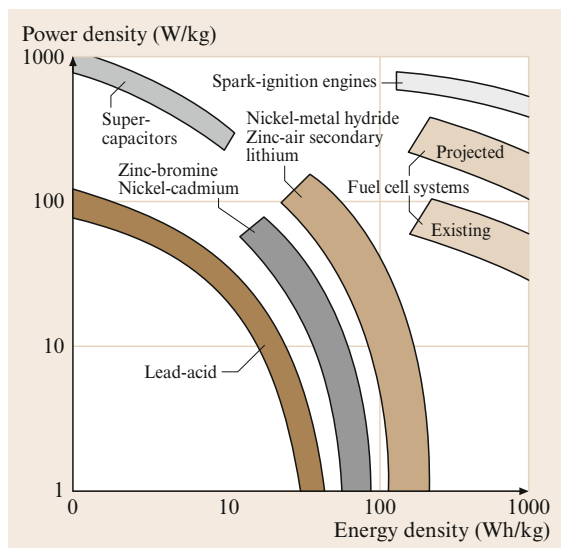


Fig. 48.26 Power and energy density for various power storage and supply systems

engines, the order of magnitude power density compared with any type of battery-powered system makes implementation feasible. If pure battery power is necessary, then considerable attention must be given to the design, including state of charge monitoring and mission provisions for recharging. In the future, emerging fuel-cell technology may provide better solutions for mobile robot power systems.

48.3.6 System Architectures for Real-Time Mission Control

Because of the intrinsic nature of hazardous environments where uncertainty and lack of geometric structure are prevalent in the task environment, shared control architectures are commonly used. The degree of human interaction runs the gamut from pure teleoperation (manual control) to high-level intelligent control operations (autonomous). For example, most low-cost EOD type robots utilize simple manual control architectures, while at the other end of the spectrum, the MARS Rover robots operate as autonomous agents responding to mission-level commands from human operators on the earth. The basic architectures for real-time mission control are summarized in Fig. 48.27 [48.73]. Also, see *Sheridan* for a comprehensive discussion of architecture principles and design [48.43]. Between the manual and autonomous control exist combinations of computer assistance and semiautonomous (i. e., selective and in situ task automation) functions intended to improve remote work performance by either reducing operator workload or allocating tasks more suitable for computer control to autonomous execution. It is important to recognize that, as one moves from manual control toward autonomous control, the data throughput rates for communication and control are reduced because high-level commands are comparatively small packets of information while force-reflection manual control of a manipulator, for example, requires visual feedback and high-bandwidth control interconnection of the slave and master controller. The data communication bandwidth available has great bearing on the type of architecture that must be used. Untethered undersea applications usually require semiautonomous architectures because of the bandwidth limitations of acoustic communications links.

As intelligent systems research advances, it is expected that, for hazardous situations, robots will employ semiautonomous and autonomous architectures to a greater degree as a fundamental approach to overall performance optimization. Operators will be able to choose the control mode based on the task needs

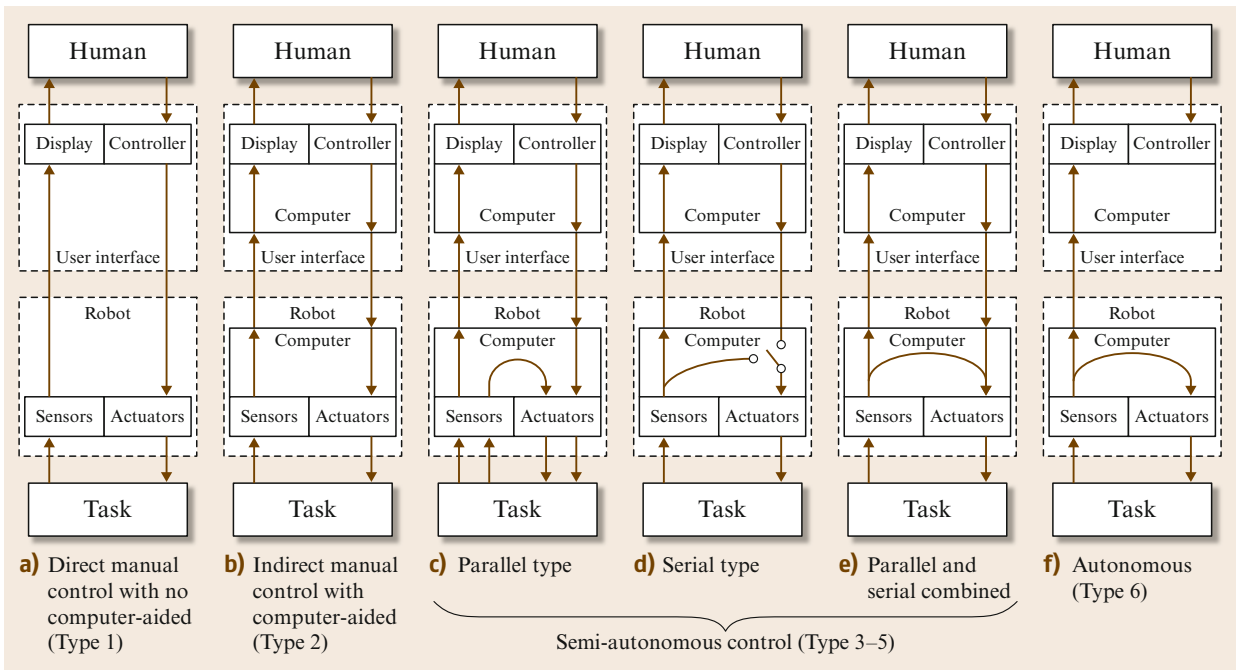


Fig. 48.27a-f System control architectures

and the likelihood that autonomous execution can be reliably achieved. In this human-centered way, task

execution mode will be matched to best achievable performance.

48.4 Conclusions and Further Reading

If we are to learn anything from the experience of the last decade we must appreciate that robotics research is just one step in the development of tools that extend human capabilities. This search for improved tools is as old as humanity itself so great patience is required [48.74].

Researchers need to make significant advances on four different fronts: mechatronics design, sensing, machine intelligence, and problem understanding.

- Mechatronics designers have to trade gains in precision, dexterity, mobility, and strength for losses in endurance and reliability. We are mostly well short of biological (e.g., human) capabilities except for high-precision applications. Machines have much the same environmental tolerance as people. Machines need special precautions against heat and cold outside the temperature range of 0–35 °C, and operation beyond –60 °C or +60 °C is usually impractical. Dust, radiation, low or zero air pressure,

fumes, biological agents, even insects can be fatal for machines without special design features, which may result in performance reductions. Maintenance or repair work may necessitate decontamination before people can work on the machinery.

- While electronic sensors can go far beyond biological capabilities, hazardous applications still present problems far beyond present capabilities as we have seen in the case of landmine applications. Radiation levels that are lethal for humans can also quickly kill electronic sensors. Extreme heat or cold limits performance, as can contamination, and even insects.
- Advances in usable machine intelligence have been much more difficult than many expected. The only real progress has been achieved with capabilities often associated with clever people: logic, mathematical manipulation, and playing games. Even the least intelligent animals have capabilities that still defy present-day understanding. However, teleoper-

ation and supervisory control can make up for this deficit in the medium term. Shortcomings in machine intelligence represent less of a barrier than the other three fronts.

- Understanding the intended application and economic factors has proved to be equally important for researchers. Early applications rely on exploiting particular opportunities with specific applications to generate confidence that inspires others to persevere with more difficult problems. Failures in understanding, particularly with economic factors, undermines confidence, and leads to disappointment and disillusioned researchers. Attempts to develop landmine clearance robots provides a useful case study that illustrates some of the difficulties in developing robots for hazardous applications.

There is no question that future teleoperators will become hybrid telerobotic systems that allow seamless transfer between manual and autonomous operations. Specific tasks will be selectively automated by operators for the purposes of enhancing quality and/or reducing

subtask execution times. Early systems will continue to incorporate high levels of human interactivity to accommodate the limitations of current intelligent systems technology. Low-cost embedded computational power will continue to track Moore's Law, making real-time execution of complex control algorithms and virtual-reality-like graphics engines (which are really useful) routine. Robust machine learning that allows human task execution skills to be captured through learning by observation will become realizable. Through micro-electrical mechanical systems (MEMS) and engineering miniaturization, all types of sensors particularly imaging sensors will be more powerful and less costly. In the next decade, telerobots for hazardous environments will become smarter and their relationship with their human operators will become more cooperative and less interactive. Human operators will supervise the operation of multiple telerobots. However, telerobots will remain subordinate and the need to assure human-takeover capabilities through teleoperation will always be present.

Readers are directed to the selection of references provided in this chapter for further reading.

References

- 48.1 S. Kang, C. Cho, J. Lee, D. Ryu, C. Park, K.-C. Shin, M. Kim: ROBHAZ-DT2: Design and integration of passive double tracked mobile manipulator system for explosive ordnance disposal. In: *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2003. Ph.D. Thesis (IEEE, Piscataway 2003)
- 48.2 A. Kron, G. Schmidt, B. Petzold, M.I. Zah, P. Hinterseer, E. Steinbach: Disposal of explosive ordnances by use of a bimanual haptic telepresence system. In: *2004 IEEE International Conference on Robotics and Automation*. 2004. Ph.D. Thesis (IEEE, Piscataway 2004)
- 48.3 W.R. Hamel, R.L. Cress: Elements of Telerobotics Necessary for Waste Clean Up Automation. In: *IEEE International Conference on Robotics and Automation*. Seoul (2001)
- 48.4 J.C. Ralston, D.W. Hainsworth, D.C. Reid, D.L. Anderson, R.J. McPhee: Recent advances in remote coal mining machine sensing, guidance, teleoperation and field robotics, *Robotica* **19**(5), 513–526 (2001)
- 48.5 GICHD: *A Study of Manual Mine Clearance*, ed. by T. Lardner (International Centre for Humanitarian Demining, Geneva 2005)
- 48.6 GICHD: *A Study of Mechanical Application in Demining* (Geneva International Centre for Humanitarian Demining, Geneva 2004)
- 48.7 J.P. Trevelyan, S. Tilli, B. Parks, H.C. Teng: Farming minefields: Economics of remediating land with moderate landmine and UXO concentrations, *Demining Technol. Inform. Forum J.* **1**(3) (2002)
- 48.8 C.G. Bruschini, B. Gross: A survey on sensor technology for landmine detection, *J. Mine Action* **2**(1) (1998), <http://maic.jmu.edu/journal/2.1/home.htm>
- 48.9 A. Göth, I.G. McLean, J.P. Trevelyan: How do dogs detect landmines? A summary of research results. In: *Mine Detection Dogs: Training, Operations and Odour Detection*, ed. by I.G. McLean (Geneva International Centre for Humanitarian Demining, Geneva 2003) pp.195–207
- 48.10 P. Debenest, E.F. Fukushima, Y. Tojo, S. Hirose: A new approach to humanitarian demining : Part 2: Development and analysis of pantographic manipulator, *Autonom. Robot.* **18**(3), 323–336 (2005)
- 48.11 S. Havlik: A modular concept of the robotic vehicle for demining operations, *Autonom. Robot.* **18**(3), 253–262 (2005)
- 48.12 J.P. Wetzel: Robotic Applications in Humanitarian Demining. In: *Engineering, Construction and Operations in Challenging Environments Earth and Science 2004: Proceedings of the Ninth Biennial ASCE Aerospace Division International Conference*. Ph.D. Thesis (American Society of Civil Engineers, Reston 2004)
- 48.13 J.P. Trevelyan: A Suspended Device for Humanitarian Demining. In: *MD96: IEE Conference on Detecting Abandoned Landmines*. (1996)

- 48.14 J. Trevelyan: Robots: a premature solution for the land mine problem. In: *Proceedings of Eighth International Symposium on Robotics Research*. (1998)
- 48.15 J.-D. Nicoud: Vehicles and robots for humanitarian demining, *Ind. Robot.* **24**(2), 164–168 (1997)
- 48.16 J.P. Trevelyan: Landmine Research: Technology Solutions Looking for Problems. In: *Symposium on Defense and Security: 5415–Detection and Remediation Technologies for Mines and Minelike Targets IX* (2004)
- 48.17 H.M. Choset, E.U. Acar, A.A. Rizzi, J.E. Luntz: Sensor-based planning: exact cellular decompositions in terms of critical points. In: *Mobile Robots XV and Telem manipulator and Telepresence Technologies VII, Proceedings of the SPIE – The International Society for Optical Engineering*. (2000)
- 48.18 Y. Zhang, M. Schervish, E.U. Acar, H. Choset: Probabilistic methods for robotic landmine search. In: *Proceedings of RSJ/IEEE International Conference on Intelligent Robots and Systems*, 29 Oct.–3 Nov. 2001. (2001)
- 48.19 S. Singh, S. Thayer: Inspired by immunity, *Nature* **415**, 468–470 (2002)
- 48.20 S. Sathyanath, F. Sahin: Application of artificial immune system based intelligent multi agent model to a mine detection problem. In: *SMC2002: IEEE International Conference on Systems, Man and Cybernetics*, 6–9 Oct. 2002. (2002)
- 48.21 D. Goldberg, M.J. Mataric: Maximizing reward in a non-stationary mobile robot environment, *Autonom. Agents Multi-Agent Syst.* **6**(3), 287–316 (2003)
- 48.22 R.A. Russell: Locating underground chemical sources by tracking chemical gradients in 3 dimensions. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 28 Sept.–2 Oct. 2004. (2004)
- 48.23 D.C. Conner, P.R. Kedrowski, C.F. Reinholtz J.S. Bay: Improved dead reckoning using caster wheel sensing on a differentially steered three-wheeled autonomous vehicle. In: *Mobile Robots XV and Telem manipulator and Telepresence Technologies VII, Proc. SPIE 2000* (2000)
- 48.24 S.-D. Kim, C.-H. Lee, S.-J. Yoon, H.-K. Jeong, S.-C. Kang, M.-S. Kim, Y.-K. Kwak: Variable configuration tracked mobile robot for demining operations. In: *2004 ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, (American Society of Mechanical Engineers, New York 2004)
- 48.25 K. Nonami, R. Yuasa, D. Waterman, S. Amano, H. Ono: Preliminary design and feasibility study of a 6-degree of freedom robot for excavation of unexploded landmine, *Autonom. Robot.* **18**(3), 293–301 (2005)
- 48.26 Y.-J. Lee, S. Hirose: Three-legged walking for fault-tolerant locomotion of demining quadruped robots, *Adv. Robot.* **16**(5), 415–426 (2002)
- 48.27 T. Wojtara, K. Nonami, H. Shao, R. Yuasa, S. Amano, D. Waterman, Y. Nobumoto: Hydraulic master-slave land mine clearance robot hand controlled by pulse modulation, *Mechatronics* **15**(5), 589–609 (2005)
- 48.28 N. Furihata, S. Hirose: Development of mine hands: extended prodder for protected demining operation, *Autonom. Robot.* **18**(3), 337–350 (2005)
- 48.29 H. Yabushita, M. Kanehama, Y. Hirata, K. Kosuge: 3D ground adaptive synthetic aperture radar for landmine detection. In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2–6 Aug. 2005. (2005)
- 48.30 A.A. Faust, R.H. Chesney, Y. Das, J.E. McFee, K.L. Russel: Canadian teleoperated landmine detection systems. Part I: the improved landmine detection project, *Int. J. Syst. Sci.* **36**(9), 511–528 (2005)
- 48.31 A.A. Faust, R.H. Chesney, Y. Das, J.E. McFee, K.L. Russel: Canadian teleoperated landmine detection systems. Part II: Antipersonnel landmine detection, *Int. J. Syst. Sci.* **36**(9), 529–543 (2005)
- 48.32 K.N. Zachery, G.M. Schultz, L.M. Collins: Force Protection Demining System (FPDS) detection subsystem. In: *Detection and Remediation Technologies for Mines and Minelike Targets X*. Ph.D. Thesis (International Society for Optical Engineering, Bellingham 2005)
- 48.33 J. Coronado-Vergara, G. Avina-Cervantes, M. Devy, C. Parra: Towards landmine detection using artificial vision. In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2–6 Aug. 2005. (2005)
- 48.34 J.P. Trevelyan: Demining Research at the University of Western Australia. (Perth 2000), <http://www.mech.uwa.edu.au/jpt/demining>
- 48.35 S. Rajasekharan, C. Kambhampati: The current opinion on the use of robots for landmine detection. In: *2003 IEEE International Conference on Robotics and Automation*. Ph.D. Thesis (Institute of Electrical and Electronics Engineers, Taipei 2003)
- 48.36 J.P. Trevelyan: Reducing Accidents in Demining (The University of Western Australia 2000), <http://www.mech.uwa.edu.au/jpt/demining/reports.html>
- 48.37 GICHD: *A Study of Manual Mine Clearance: Part 4 – Risk Assessment and Risk Management*, ed. by T. Lardner (Geneva International Centre for Humanitarian Demining, Geneva 2005)
- 48.38 P. Newnham, D. Daniels: The market for advanced humanitarian mine detectors. In: *Detection and Remediation Technologies for Mines and Minelike Targets VI*. Ph.D. Thesis (The International Society for Optical Engineering, Orlando 2001)
- 48.39 J. Vertut, P. Coiffet: Teleoperation and robotics, evolution and development, *Robot Technol.* **3A**, 302–307 (1985)

- 48.40 L.I. Slutski: *Remote Manipulation Systems: Quality Evaluation and Improvement* (Kluwer Academic, Dordrecht 1998)
- 48.41 K.F. Kraiss: *Advanced Man-Machine Interaction: Fundamentals and Implementation* (Springer, Berlin, Heidelberg 2006)
- 48.42 E.G. Johnsen, W.R. Corliss: *Human Factors Applications in Teleoperator Design and Operation* (Wiley, New York 1971)
- 48.43 T.B. Sheridan: *Telerobotics, Automation, and Human Supervisory Control, Technology and Industrial Arts* (MIT Press 1992) p. 432
- 48.44 W.R. Hamel: Adapted from Chapter 10, Sensor-Based Planning and Control in Telerobotics. In: *Control in Robotics and Automation, Sensor-Based Integration*, ed. by B.K. Ghosh, N. Xi, T.J. Tarn (Academic, New York 1999) pp. 285–309
- 48.45 J.V. Draper: Effects of Force Reflection on Servomanipulator Performance. In: *1987 International Topical Meeting on Robotics and Remote Handling in Hostile Environments* (1987)
- 48.46 R. Goertz: Manipulator System Development at ANL. In: *12th Conference on Remote Systems Technology* (1962)
- 48.47 R. Goertz: Some Work on Manipulator Systems at ANL, Past, Present and a Look at the Future. In: *1964 Seminars on Remotely Operated Special Equipment* (1964)
- 48.48 D.a.M., H. L. Kuban: An Advanced Remotely Maintainable Servomanipulator Concept. In: *1984 National Topical Meeting on Robotics and Remote Handling in Hostile Environments* (Gatlinburg 1984)
- 48.49 D. Sands: Cost effective robotics in the nuclear industry, *Ind. Robot* **33**(3), 170–173 (2006)
- 48.50 S. Sanders: Remote operations for fusion using teleoperation, *Ind. Robot* **33**(3), 174–177 (2006)
- 48.51 B.L. Luk, K.P. Liu, A.A. Collie, D.S. Cooke, S. Chen: Teleoperated climbing and mobile service robots for remote inspection and maintenance in nuclear industry, *Ind. Robot* **33**(3), 194–204 (2006)
- 48.52 P. Desbats, F. Geffard, G. Piolain, A. Coudray: Force-feedback teleoperation of an industrial robot in a nuclear spent fuel reprocessing plant, *Ind. Robot* **33**(3), 178–186 (2006)
- 48.53 B. De Jong, E. Faulring, J.E. Colgate, M. Peshkin, H. Kang, Y.S. Park, T. Ewing: Lessons learned from a novel teleoperation testbed, *Ind. Robot* **33**(3), 187–193 (2006)
- 48.54 S. Hirose, H. Kuwahara, Y. Wakabayashi, N. Yoshioka: The Mobility Design Concepts/Characteristics and Ground Testing of an Offset-Wheel Rover Vehicle. In: *International Conference On Mobile Planetary Robots and Rover Roundup* (1997)
- 48.55 T. Kagiwada: Robot Design for Stair Navigation, *Jpn. Soc. Mech. Eng. Int. J. Series C* **39**(3), 629–635 (1996)
- 48.56 R. Volpe, R. Ohm, R. Petras, J. Welch, J. Blaram, R. Ivlev: A Prototype Manipulation System for Mars Rover. In: *IEEE/RSJ International Conference on intelligent Robots and Systems* (Grenoble 1997)
- 48.57 R. Siegwart, P. Lamon, T. Estier, M. Lauria, R. Piguët: Innovative design for wheeled locomotion in rough terrain, *Robot. Autonom. Syst.* **40**, 151–162 (2002)
- 48.58 K. Yoneda, Y. Ota, S. Hirose: Development of a Hi-Grip Stair Climbing Crawler with Hysteresis Compliant Blocks. In: *4th International Conference on Climbing and Walking Robots (CLAWAR 2001)* (Karlsruhe 2001)
- 48.59 S. Hirose, T. Sensu, S. Aoki: The TAQT Carrier: A Practical Terrain-Adaptive Quadru-Track Carrier Robot. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems* (1992)
- 48.60 H. Schempf, E. Mutschler, C. Piepgras, J. Warwick, B. Chemel, S. Boehmke, W. Crowley, R. Fuchs, J. Guyot: Pandora: Autonomous Urban Robotic Reconnaissance System. In: *International Conference on Robotics and Automation* (Detroit 1999)
- 48.61 J.B. Coughlan: *Small All Terrain Mobile Robot* (Remo Tec, USA 1991)
- 48.62 T. Takayama, S. Hirose: Development of Souryu-I connected crawler vehicle for inspection of narrow and winding space. In: *26th Annual Conference of the IEEE Industrial Electronics Society* (Nagoya 2000)
- 48.63 C.H. Cho, W.S. Lee, S. Kang, M.S. Kim, J.B. Song: Rough-Terrain Negotiable Mobile Platform with Passively Adaptive Double-Tracks and its Application to Rescue Missions, *Adv. Robot.* **19**(4), 459–475 (2005)
- 48.64 K. Kato, S. Hirose: Development of the quadruped walking robot, TITAN-IX-mechanical design concept and application for the humanitarian demining robot, *Adv. Robot.* **15**(2), 191–204 (2001)
- 48.65 iRobot Inc. 2006. <http://www.irobot.com/> May 21
- 48.66 E. Cheung, V.J. Lumelsky: Proximity sensing in robot manipulator motion planning: System and implementation issues, *IEEE Trans. Robot. Autom.* **5**, 740–751 (1989)
- 48.67 B.R. Shetty, M.H. Ang Jr.: Active compliance control of a PUMA 560 robot. In: *IEEE International Conference on Robotics and Automation* (1996)
- 48.68 B.-S. Kim, S.-K. Yun, S. Kang, C.-S. Hwang, M.-S. Kim, J.-B. Song: Development of a joint torque sensor fully integrated with an actuator. In: *2005 International Conference on Control, Automation and Systems* (KINTEX, Gyeonggi-Do 2005)
- 48.69 M. Uebel, M.S. Ali, I. Minis: *The Effect of Bandwidth on Telerobot System Performance* (Goddard Spaceflight Center 1991)
- 48.70 G. Niemeyer: Telemanipulation with Time Delays, *Int. J. Robot. Res.* **23**(9), 873–890 (2004)
- 48.71 D. Ryu, S. Kang, M. Kim, J.-B. Song: Multi-modal user interface for teleoperation of ROBHAZ-DT2

field robot system. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 28 Sept.–2 Oct. 2004 (2004)

48.72 A.K. Skula, A.S. Arico, V. Antonucci: An appraisal of electric automobile power sources, *Renewab. Sustainab. Energy Rev.* **5**, 137–155 (2001)

48.73 K.W. Ong, G. Seet, S.K. Sim: Sharing and trading in a human–robot system. In: *Cutting Edge Robotics*, ed. by V. Kordic, A. Lazinica, M. Merdan (2005) pp. 467–496

48.74 J. Trevelyan: Redefining robotics for the new millennium, *Int. J. Robot. Res.* **18**(12), 1211–1223 (1999)

Mining Robotics

Peter Corke, Jonathan Roberts, Jock Cunningham, David Hainsworth

Mining is the process of extracting mineral resources from the Earth for commercial value. It is an ancient human activity which can be traced back to Palaeolithic times (43 000 years ago), where for example the mineral hematite was mined to produce the red pigment ochre. The importance of many mined minerals is reflected in the names of the major milestones in human civilizations: the stone, copper, bronze, and iron ages. Much later coal provided the energy that was critical to the industrial revolution and still underpins modern society, creating 38% of world energy generation today. Ancient mines used human and later animal labor and broke rock using stone tools, heat, and water, and later iron tools. Today's mines are heavily mechanized with large diesel and electrically powered vehicles, and rock is broken with explosives or rock cutting machines (Fig. 49.1).

49.1 Background	1127
49.1.1 Robotics in Mining	1129
49.2 Metalliferous Mining	1132
49.2.1 Underground Hard-Rock Metalliferous	1132
49.2.2 Scope for Robotics	1135
49.2.3 Case Study: Nonconventional Robotic Miner	1139
49.3 Underground Coal Mining	1140
49.3.1 Coal Mining Process	1141
49.3.2 Scope for Robotics	1142
49.3.3 Rapid Roadway Development	1143
49.4 Surface Coal Mining	1144
49.4.1 Dragline Excavator	1144
49.4.2 Truck-Shovel Operation	1146
49.4.3 Surface Haulage	1147
49.5 Conclusions and Further Reading	1148
References	1148

Mining remains hard and hazardous work and the scope for the application of robotics is very high. Mining requires the handling of enormous quantities of material in a cost-effective and safe manner. High operational costs, the need for greater productivity and improved health and safety outcomes are powerful drivers for robotics. Existing approaches to improving productivity

and safety through training, work practices, and larger and improved machine design are providing diminishing returns. The industry is close now to a tipping point where robotics and automation will provide the next step change in productivity and safety. A review of the global status of robotics in operating mines as of 2006 can be obtained from [49.1].

49.1 Background

For the purposes of this chapter, mining may include minerals, coal, salts, gemstones, tar-sands, shale oil, and construction material, which we will refer to collectively as commodities. Commodities may be exposed at or close to the surface, or deep underground. They may be concentrated into seams or veins or evenly dispersed

so that a significant amount of waste material must also be extracted and dealt with. Economic black coal is always found as a monolithic seam, a consequence of the manner in which it is formed by vegetable matter accumulating on the floor of ancient seas and lakes. Gold can be evenly disseminated or found in seams of quartz that

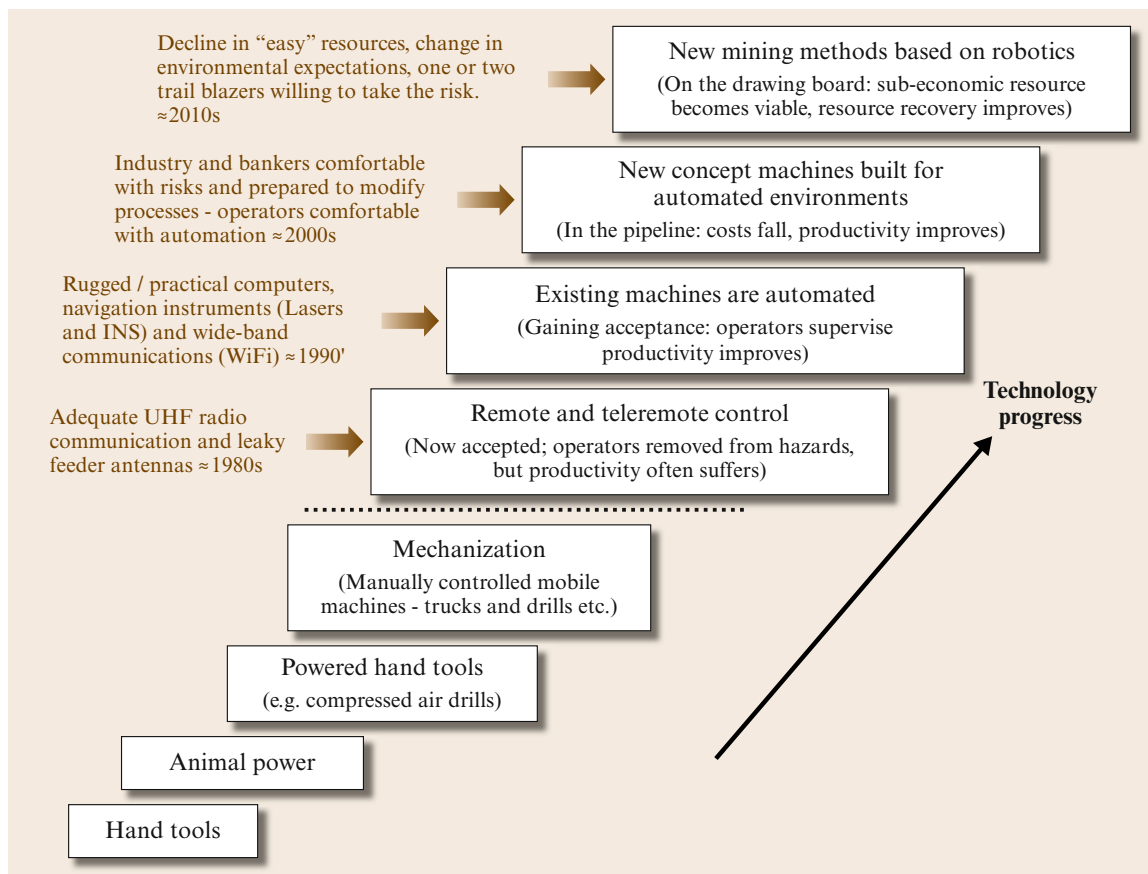


Fig. 49.1 Evolution of mining technology

was long ago forced through fissures in the surrounding rock mass. Some mineral deposits such as Canada's Sudbury nickel, are speculated to have come from me-

eteorite impact so the resource is distributed around what seems to be an extremely large crater. The extraction of petroleum and natural gas is not generally considered as

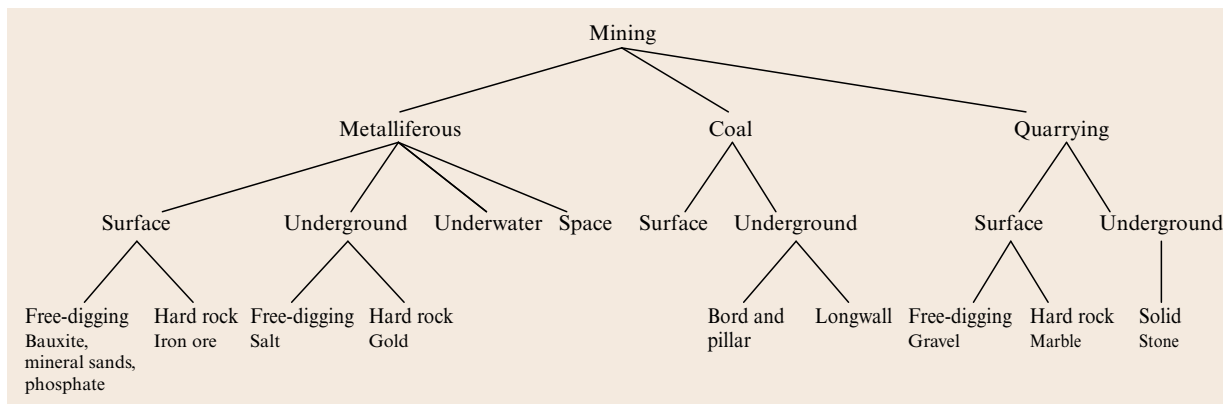


Fig. 49.2 A taxonomy of mining

mining and will not be discussed in this chapter. Mining may occur on land, below the surface, on the ocean floor, and in the future in space. A taxonomy of mining is presented in Fig. 49.2 as a framework for discussion. The first row distinguishes the different broad class of material that is extracted. Metalliferous refers to metal-bearing ores. Coal is found as black coal, a hard black rock-like material, or softer wetter brown coal. Quarrying refers to the extraction of stone, gravel or sand which is used as a building material or for use in concrete and road base.

The second row in the taxonomy differentiates the location of the material. Those on the surface of the Earth are extracted using large open-pit operations such as shown in Fig. 49.3. Underground mining methods are more expensive but are used for deeper material and this requires excavation of shafts and tunnels. Mining of deep ocean trench, black smoker sulphide resources is just beginning. The grade, or mineral concentration, of these resources is very high by terrestrial standards and mining methods are just being developed. The cost of reaching these resources is extremely high, and concern about fragile underwater ecosystems has acted as a brake on their exploitation.

The third row differentiates the physical state of the resource. Free digging materials such as bauxite, mineral sands, oil shales or salt can be simply *scooped up*. Harder materials need to be fragmented, typically by placing explosives in holes drilled into the ore. Rock cutting methods are sometimes used but this is applicable to only a small range of materials and the cost of cutting tools and power to drive the cutting process are often limiting factors. Development in this area is driven primarily by advances in hard materials for the cutting teeth.



Fig. 49.3 An open cut coal mine with a dragline excavator (US Bureau of Land Management, Washington)

A marked difference between current techniques for free-digging materials such as coal and hard rock (metalliferous) mining is the continuous nature of the former compared to the cyclical nature of the latter. Soft material can be extracted and conveyed to the surface continuously, whereas in the case of hard rock mining the process is generally not continuous, requiring breakage through blasting then loading and hauling on a cyclic basis.

49.1.1 Robotics in Mining

In this chapter we distinguish between several related technologies: automation, teleoperation and robotics. We define automation as conventional computer- or programmable logic controller (PLC)-based control systems, and teleoperation as remote control of equipment by an operator using information provided by video cameras and graphical display of machine state. This section discusses various aspects related to the introduction of robotics in mining.

Differences from Factory Robotics

Existing mining equipment is typically hydraulically actuated and either diesel or electrically powered. Work tools or end-effectors are often located at some distance from the operator and power source. Loads and masses are high and the equipment is expensive. In robotic terms, equipment is heavy, compliant and mobile. Localization, particularly absolute localization, is extremely difficult. The operating environment may also be extreme, in terms of high temperatures, salty and acidic water, and exposure to rockfall and robust treatment. The equipment may also be hazardous to humans, and in the transition period before fully robotic mines, the issue of human safety is paramount. Current prototype mining robots are essentially converted mining machines, but over time they will evolve to be purpose-built robots.

Acceptance by Mine Owners

To date, there have been few successful and practical robotic applications that have met the difficult operational requirements. There are some good reasons for this. Unlike traditional areas that have high uptake of robotics/automation such as factories and even mineral processing plants, the mining environment is not stable and is in a constant state of evolution. Machines and people must be adaptive and flexible and engage the natural terrain where they find it. Mining environments are difficult to alter to suit the easy application of automated machines. The design and configuration of factory and

process machinery can easily be optimized and integrated into a well-performing and stable system. Before finance can be raised to establish a mine, the resource needs to be well defined and valued. This is essentially a geoscience-intensive statistical process resulting in some uncertainty. This may leave little tolerance to accept risk in the downstream activities such as mine design, construction, and equipment selection which are predominantly engineering focused. Mine designers and ultimately investors like to use data and experience from existing successful operations. Therefore, any introduction of technology, including new mining methods is likely to be incremental, evolutionary, and undertaken with minimal disruption to existing production.

Many in the mining industry have little familiarity with robotics or its potential. Mining engineering is a combination of mechanical engineering, geomechanics, geology, finance, and management. It is critical that other disciplines work closely with production-focused mining engineering staff so that the technology will be fit for purpose and/or the uptake will not be inhibited through a lack of understanding of business needs. Significant progress is often made by adapting technology developed for other markets and disciplines.

The Need for Integration

When a mine is in operation, there can be many unexpected conditions that present challenges to conventional robots that excel in well-controlled environments. The inability to manage exceptions will usually negate any superior performance that might be expected based on operation under controlled conditions.

Robotic machines must be integrated with manual operations and their effects on up- and downstream processes must be properly considered, for example, if an automated machine requires an isolated area for safety reasons, and the same area requires regular or periodic access by people, then the benefits may be lost. To capture the benefits, other mining processes need to be modified to suit and a *systems* approach is recommended.

Remote control is often a good first step towards automation. The use of remote control either within line of sight or with the aid of video cameras/sensors (teleoperation) is now accepted by the mining industry, although it has yet to achieve market saturation. At the time of writing, a range of existing machines are being converted to *robots* through the application of automation technologies. Most notably these include highwall mining machines, load haul dump vehicles (LHDs), rock drills, and haulage trucks.

There are few suppliers of this specialized equipment and they are experiencing significant demand beyond their production capability. Robotic conversion of draglines and coal shearers is now possible and prototype systems are being installed. New concept machines, such as for the robotic installation of ground support and explosives, are at prototype stage or undergoing early field trials. New mining methods that are designed from the outset for robotic machines (see, for example, Sects. 49.2.3 and 49.3.3) are currently on the drawing board waiting for finance and trials.

The wide diversity of mining operations, methods, and environments will also be discussed because these affect the selection, design, and limitations of robotic applications.

Productivity

Productivity gains from consistency of operation, rather than an increase in basic production rates, is a priority. Because of the inability of current automated or robotic mining systems to match the sensing and control functions of operators, such systems are unlikely to match the peak performance of the best operators. However the strength of automation systems is their ability to maintain consistent equipment operation over extended periods of time. In an essentially continuous process this brings real benefits. Moreover, equipment operation can be constrained to always lie within performance bounds, conferring benefits in enhanced equipment reliability and improved product quality translating to availability and thence to productivity.

The largely batch nature of metalliferous mining and the time taken to change shifts often provides an opportunity to automate equipment so that it operates continuously during meal breaks and shift changes. Typically, two to three hours can be saved each shift in this way, especially if machines can be controlled from a centrally located control room. Access to the machine can then be limited to regular servicing or relocation. Hot-seat changeover is also used to address this issue, although this often results in increased labor costs and longer or overlapping shifts. In the case of LHDs, improved productivity of 20% has been experienced through automation.

Health and Safety

When considering safety, it is self-evident that there are potential benefits if people can be removed from the hazardous environments near a mining face or equipment. Here, injuries from explosive atmospheres, noise, dust, rockfalls, and moving equipment are commonplace.

In China, there have been more than 5000 fatalities per year in the coal mining industry, although the numbers are falling steadily. Figure 49.4 shows the declining number of fatalities in the Australian mining industry from statistics published by the Minerals Council of Australia (MCA). Further examination of this data identifies two categories of fatal injury that are significantly higher than all other categories: *vehicles or mobile plant* (31%) and *slides, cave-ins or rock-falls* (25%). When some of the other equipment-related incidents involving entrapment or machine faults are included, the *vehicle and mobile plant* category increased to approximately 36% of mining deaths over that period. The third most significant category relates to *falls, slips, and trips* (10.5%). Similar category distributions were found in the **US** mining and quarrying industry (National Institute for Occupational Health and Safety (**NIOSH**) statistics). It is obvious that people cannot be hurt by mobile equipment if they are not close by. Hence removal of operators from the area using robotic technology with compatible procedural controls is a viable solution.

In both coal and metalliferous underground mining, access tunnels must be supported by one or more methods including rock bolts, sprayed concrete, and steel mesh. There is always a period of time and location where ground support has not been installed following blasting or cutting and there is a risk of rockfall. Installation of ground support involves manual and mechanized activities and is a hazardous task. According to Potvin et al. [49.2] there has been a significant reduction in injuries from metalliferous mine rockfalls since 1998. From Potvin's report, mining procedures and training have played a significant role in the reduction. However, the majority of rockfall deaths still occur in access tunnels (drives) less than 10m from an active mining face. Two-thirds of these deaths occurred during the installation of ground support or explosives charging. Again the logical way of reducing or avoiding fatalities and injuries is by removal of operators from the immediate vicinity where possible.

In order to make a significant further reduction in the number of mining fatalities and injuries without changing mining methods, the following areas need to be addressed as a priority with the introduction of robotic technology and proximity protection:

- Mobile equipment (LHDs, trucks, longwall equipment)
- ground support installation processes (including scaling and coal mine roadway development)

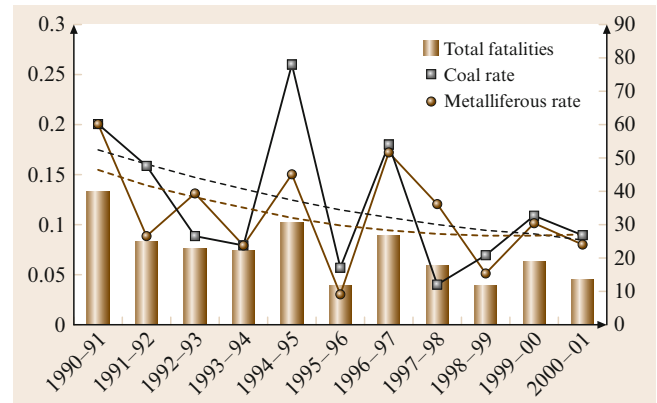


Fig. 49.4 History of fatalities in the Australian mining industry. Left x-axis is Fatalities per 1M man hours. Right x-axis is number of fatalities

- explosive placement
- rock drilling

It also needs to be acknowledged that some early introduction of close-range remote control and automation of existing mining equipment has actually resulted in deaths and injury. This was largely caused by a heavy reliance on procedural controls to maintain isolation from people. Operators were killed or injured by accidentally running over themselves or being caught in machines during maintenance. Poor design of control systems and interlocks may also have contributed to these incidences and has led to the development of industry standards. Failsafe engineering controls are critical to any robotic installation.

Acceptance by Miners

Despite the clear health and safety benefits there are perceptions in the workplace that these technologies will impact adversely on job security. The effect of this should not be underestimated but it can be managed. Consequently the current thrust of robotics and automation development in mining is to employ these systems to remove operators from direct roles in the most hazardous operational areas to safe-area supervisory and backup involvement. The robotics challenge is to develop the sensors necessary to replace the functions currently carried out by operators in the hostile operating environment, making their work easier.

There are many undesirable, monotonous, and hazardous tasks that can be addressed as a priority with general acceptance of the workforce if correctly introduced. Removing people from these activities to

a role that involves supervision of a process instead of moment-by-moment operation will generally be well received by operators because it makes their work easier and safer. Robotics also enables a reduction in the mining workforce as productivity improves and this can cause resentment in some mining communities. This can be managed through natural attrition as the workforce ages and people are less inclined to seek employment in the industry. Over the longer term the skill base will change and new jobs will be

created in knowledge-intensive areas associated with development and maintenance of robotic equipment. In Western countries with ageing populations, the shortage of skilled workers willing to live the hard and often remote lifestyle will drive robotics. Robotic mining could also keep mines open longer when it becomes uneconomic to mine using existing methods, mirroring the effect observed in the automotive industry where robotics has kept factories open rather than displacing workers.

49.2 Metalliferous Mining

For the purpose of this chapter, *metalliferous* is defined as all mined materials apart from coal, oil, and gas; for example, targeted materials may include, copper, lead, zinc, silver, gold, diamonds, iron, and aluminium. Several mining methods are used to recover metalliferous ore and references can be consulted for more detail about each [49.3–5]. Mining methods can be broadly categorized into surface and underground. Variations on the methods relate to the type of ore body or rock rather than the commodity or mineral they hold. For example, components of interest can be *disseminated*, that is distributed relatively evenly through a large volume of terrain. Here the boundaries of the ore body must be defined by the economics of mining. Such ore bodies are typically extracted using *mass mining* techniques. Ore can also be distributed in veins or tabular shapes ranging from a few centimeters to many meters thick. The boundaries of these ore bodies are typically defined by geology and they favor methods that are more selective, especially for underground mining.

Ore types can also be defined by hardness and the stability of the rock mass. Hard-rock ores usually require *drilling and blasting*. In some circumstances they can be made to cave-in if a large enough slot of rock is removed from below the volume of ore. Unconsolidated materials such as phosphate, salt, mineral sands, and bauxite are usually mined using excavators, rippers, shovels, and even dredges. Little or no explosives are used except where the ore is capped by hard, barren rock. These ore bodies are free digging and often alluvial in origin. Mineral sand is often mined using dredgers that float on an artificially created pond in the mineral zone. Sand is sucked up by a dredge and pumped to a processing plant. Another example of alluvial ore is presented by surf-zone marine deposits of diamonds which are located off

the southern coast of Africa. Here, the heavier diamonds are separated from sand through wave action and they become trapped in rock crevasses under the surf zone of a beach. Such deposits are mined using what is essentially a giant vacuum cleaner.

The following discussion of mining methods and infrastructure is made with the emphasis on aspects that are important for the design and application of robotics.

49.2.1 Underground Hard-Rock Metalliferous

The cost of underground mining can be between two and ten times greater than for surface mining. Therefore ore needs to be relatively rich to be economically viable for underground mining.

With some exceptions, metalliferous mining is a batch process involving a sequence of physical operations including:

- exploratory drilling
- geological and geotechnical mapping
- developing access roadways or tunnels (development)
- drilling and blasting of areas containing the ore (*stopes*)
- extraction and haulage of the broken ore
- back fill of extracted zones and rehabilitation

There are a large variety of different metalliferous mining methods that will weigh these separate process steps differently. Each has different opportunities for automation and robotics. Metalliferous mining has a significant reliance on explosives. However, new rock cutting technology is now emerging that will see a shift

away from explosives, at least in the development of access roadways and for some surface mining [49.6]. Rock cutting applications will benefit from a high level of robotic control. In metalliferous mining there is a strong emphasis on total extraction of the resource, which often means that voids left by mining must be filled to ensure geological stability.

Figure 49.5 shows one example of an underground mine layout. Here, the mine has progressed from a surface mine to an underground mine as the easily available ore is depleted (e.g., the Kiruna mine in Sweden). In some cases, underground mines can also evolve into surface mines (e.g., the Kalgoorlie Consolidated Gold Mine in Australia).

Layout and Environment

Access to underground is either by a declining tunnel (decline) or shaft or sometimes both. A decline is a descending tunnel with a typical gradient of 1:7 which is also known as a ramp. Cross-sectional dimensions are typically between 5 m and 6 m and the road surfaces consist of compacted gravel, often mixed with finer material such as tailings to assist with consolidation. Road maintenance is a critical task that significantly affects the productivity of heavy vehicles. The Kiruna iron mine in Sweden is an exception because it has a bitumen decline roadway that is suitable for normal highway vehicles. A typical decline will consist of a series of straight sections between 200 m and 400 m long which are joined with a relatively tight 180° spiral turn to keep the tunnel in close proximity to the ore body. In some mines the decline is a continuous spiral for its whole length. Exits are placed periodically to give access to mine working levels, which consist of a network of tunnels. Tunnels are often called *drives* where they run roughly inline with the ore body and *crosscuts* where they cut across the ore body. The terminology is not strictly defined. Usually the roadways are one-way and contain periodic passing bays. The traffic is mixed, with light four-wheel-drive (4WD) service or utility vehicles operating together with large haul trucks that may have a gross vehicle mass of up to 110 t.

Virgin rock temperatures can reach 80°C, so operating areas including stopes are ventilated using air pumped from the surface. Sometimes air must be cooled and this is usually done near the operating areas using water that has been chilled at the surface and piped underground. Ventilation circuits are well managed and monitored to ensure operating areas are not downstream of any dusty, radioactive or fume-laden areas. This repre-

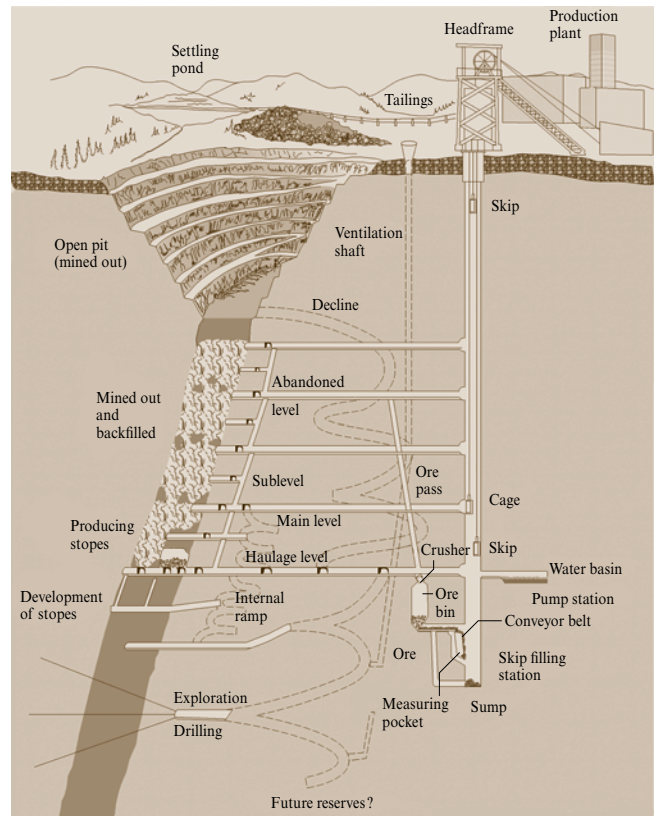


Fig. 49.5 Typical hardrock mine layout also showing transition between surface and underground mining [49.5]

sents a significant cost which can be mitigated through the use of robotics that can operate in poorer quality atmospheres.

Often mine workings intersect aquifers, giving rise to wet areas that must be drained. Mine water is often more salty than sea water and can also be extremely acidic. Electronic enclosures must be well sealed or drained and sometimes cooled. Developers also need to be aware that enclosures that are sealed in humid conditions may experience internal condensation if subsequently exposed to cooler conditions.

Block Cave Mines

The block cave underground mining method (Fig. 49.6) is the lowest cost underground method because once in production, the rock collapses and fragments under its own weight [49.5]. This minimizes the amount of explosives, equipment, and unit operations. Typically the ore bodies are large, disseminated, highly fractured or with low-tensile-strength rock. They may take a num-

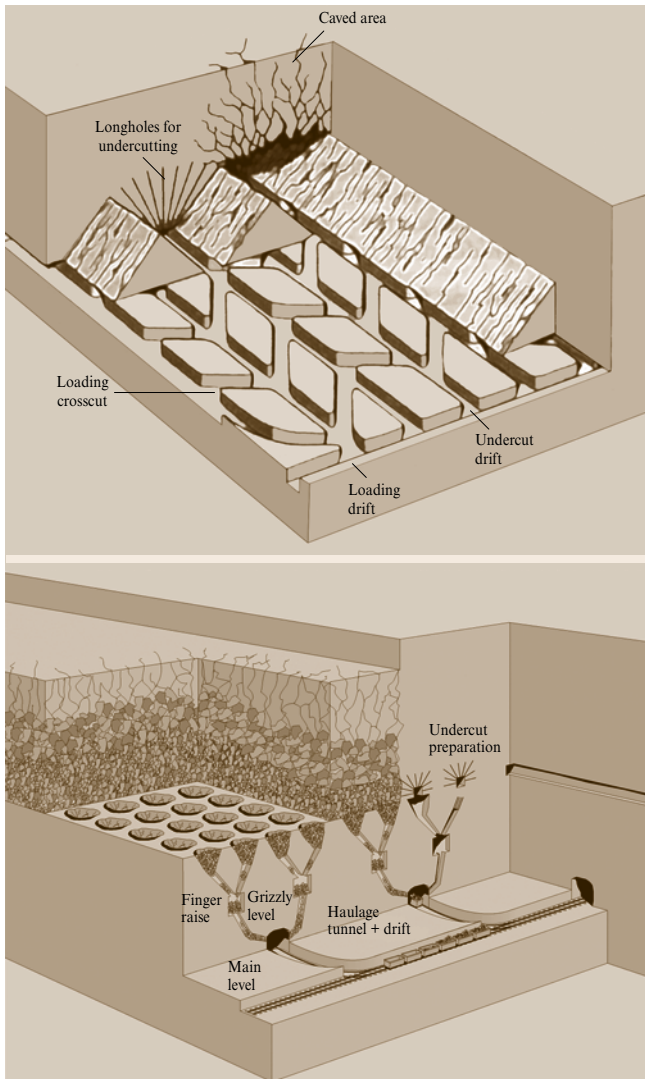


Fig. 49.6 Typical block cave underground mine

ber of years to extract, so a high level of infrastructure can be justified. This method approaches a continuous operation and is well suited to automation and robotics.

To create the cave, a large horizontal slab of ore called an *undercut* is removed at the base of the ore. The virgin rock can then no longer support itself and it caves. The broken ore *flows* downward and continues to fracture until it reaches a series of draw points which are linked to a tunnel network. Underground loaders (LHDs) pick up the ore from the draw point and take it to an underground crusher. There are several activ-

ities in block caving that may be suitable for robotics are

- automated loading and haulage
- ore hang-up clearance (including blasting)
- survey and inspection
- secondary rock breakage to clear large rocks

Stoping and Sublevel Caving

There are several other commonly used mining methods that have unique characteristics that differentiate them from a mining engineering perspective. However, for the application of robotics, they can be grouped together because they share similar challenges and equipment. These methods include:

- room and pillar
- narrow vein
- vertical crater retreat
- sublevel caving
- sublevel open stoping
- cut and fill

A classic example of a sublevel stoping method is shown in Fig. 49.7. Examples of the other methods can be found in [49.3–5].

For these methods, access is gained to the ore body at various levels and the ore is fragmented, either entirely by drill-and-blast or in combination with caving (sublevel caving). These methods involve drilling holes for explosives and loading the explosives. Unlike block caving, these other mining methods are highly mobile, infrastructure is short term, congestion is heavy, and through traffic is common in the active areas. Careful consideration is required for isolation of any robot from manual traffic.

Activities similar to those mentioned for block caving are also suitable for robotic application with the addition of production and development drilling and ground support. For these mining methods there is a significant amount of tunnel (roadway) development, which is often a rate-determining step. The activities involved are all undertaken close to an active face and are therefore hazardous. This is a cyclic process with different heavy equipment continually deployed and replaced as the tunnel progresses.

Communication Infrastructure

Underground radio communication is challenging because of the wide variation in rock conductivity

and the geometry of mine voids. Radio communication propagates with high attenuation through tunnels and voids and is significantly affected by vehicles and steel infrastructure. Currently the most common form of communication remains very high-frequency (VHF)/ultrahigh-frequency (UHF) radio which is propagated through *leaky feeder* coaxial antennas throughout the mine workings. VHF/UHF radio is used mainly for voice but also for serial data communications. For equipment teleoperation, UHF systems can also carry analog video on amplitude-modulated (AM) frequencies. Frequency-modulated (FM) video is rare because of the cost and the unusual effects caused by the highly dispersive nature of the mine terrain. If an unfortunate choice of wavelength is made, the tunnels and voids may act as waveguides and filters. This produces a complex and unstable performance for both AM and FM that can result in signal strength variation of 60 dB over tens of centimeters and significant interference from vehicles and other infrastructure.

With the relatively recent evolution of Ethernet and WiFi technology [49.7], such systems are gaining popularity and will soon be common. WiFi systems are now in common use by advanced mines and they usually include a fibre-optic trunk link from the surface to distributed access points underground. Because of the Ethernet protocol, radio propagation anomalies result in reduced and variable data rates and communication is less deterministic compared with analog communication. This variable and significant latency means that caution is required when using WiFi for control and alternate and independent safety systems are thus required. Radio transmitters are usually power limited or are otherwise isolated to avoid accidental ignition of detonators and other explosives. WiFi systems suitable for safety-critical control applications are emerging.

Through-rock radio communication is also available but it can only support data rates of tens of characters per minute in one direction (base to field). This mode of communication modulates low-frequency electromagnetic fields (a few kHz) generated from a long loop antenna located near or around production areas. The antennas carry high current at low voltage and communication shadows often exist near highly conductive ore. This communication mode can be used for issuing simple messages to personnel or simple control commands (off/on) to equipment. New technology for bidirectional through-the-rock communication is under development.

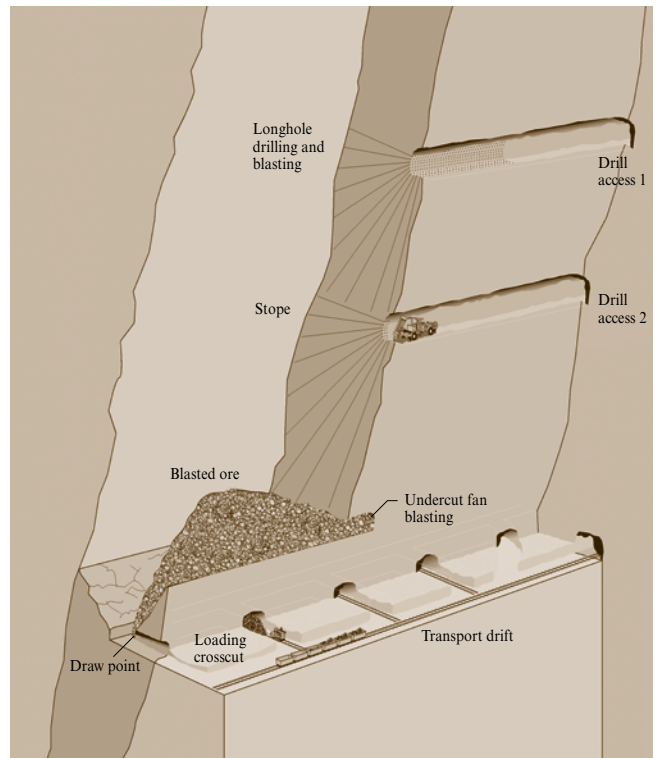


Fig. 49.7 Typical underground mining method called *sub-level stoping* (after Atlas Copco, www.AtlasCopco.com.au)

49.2.2 Scope for Robotics

This section will review aspects of underground metalliferous mining with a focus on the opportunity for, or impact on, robotics and automation.

Underground Transport

In all types of underground metalliferous mining, material must be moved from where it is mined to the surface. The movement of the material can be achieved using either conveyors, haulage vehicles or hoisted bins (*skips*). Conveyors and hoists have been largely automated using factory automation technologies.

Mobile vehicles are used to carry ore out of the mine along the declines, or to move ore from the face to loading points underground. Underground haul trucks are typically diesel powered and ascend at about 6–10 km/h, fully laden, and descend at about 20 km/h. Electric haul trucks are rarer and use pantographs to tap overhead power and are usually faster than diesel trucks, ascending at 18 km/h and descending at 24 km/h. Mobile haulage vehicles are normally used where a high



Fig. 49.8 A load haul dump (LHD) vehicle

level of flexibility is required and in particular at a mining face.

For almost all underground metalliferous mining, load-haul-dump (LHD) vehicles are used in the ore transport process. LHDs (Fig. 49.8) are configured like front-end-loaders with a large bucket but are articulated so that they can manoeuvre in relatively small spaces and through tight corners. The vehicles are *trammed* between the pick-up and dump points, which could either be a temporary stock pile, crusher, chute, shaft or haul truck. LHDs are rarely used to carry material for a distance beyond 600 m. Their maximum speed is ap-

proximately 20 km/h and they may have a clearance of less than 1 m to the tunnel wall either side. This is a tedious and hazardous task for manual operation. These vehicles can be diesel or electrically (via a trailing cable and onboard winder) powered.

The two obvious motivations for automating haulage vehicles are improved efficiency and safety. Improved efficiency can result from reducing labor costs, increasing operating duty, reducing wear and hence maintenance costs. Automation can be approached in a staged manner with each level of sophistication bringing benefits as shown in Fig. 49.9. The operation of LHDs is hazardous because the vehicle must venture into areas that are potentially unstable with a relatively high chance of rockfall. Traditionally LHDs are manually operated for the tramming cycle but operated by line-of-sight remote control while they operate in hazardous areas. Regularly changing the operating mode is time consuming and hazardous. Accidents have occurred where an operator has been crushed by the LHD as it returns from the stope by remote control. To avoid this risk, many mines now use teleoperation for the entire tramming cycle [49.8, 9]. However, this typically leads to slower tramming times than if the operator was onboard, which is due to the reduced perception and lack of feedback to operators that naturally occur when they are no longer on the vehicle. Hence, onboard vehicle navigation is the obvious solution to remove the operator from the hazard and maintain high-tramming speeds and productivity.

The automation of LHDs has been investigated by a number of researchers over the past two decades. Early systems used buried wires in concrete roadways as guidance infrastructure for LHDs [49.10]. However, concrete roadways are not common in metal mining operations and so this solution is of limited use. A common approach was to use an optical marker placed on the center line of the roof along the length of the tunnel. An upwards-facing camera was then used to track the lights and estimate the offset of the vehicle, which was fed into a steering controller. Early systems used either a painted white line [49.11] or retroreflective tape [49.12, 13]. Another system used a row of lights suspended from the centerline of the tunnel roof [49.14, 15].

Following painted lines, lines of retroreflective tape, lines of lights or wires buried in the ground has its limitations. The lack of longitudinal information (along track data) means that the speed of the LHD must remain low as the vehicle does not know when a corner is coming up. Solutions to this are to provide additional means of measuring along-track position, which could

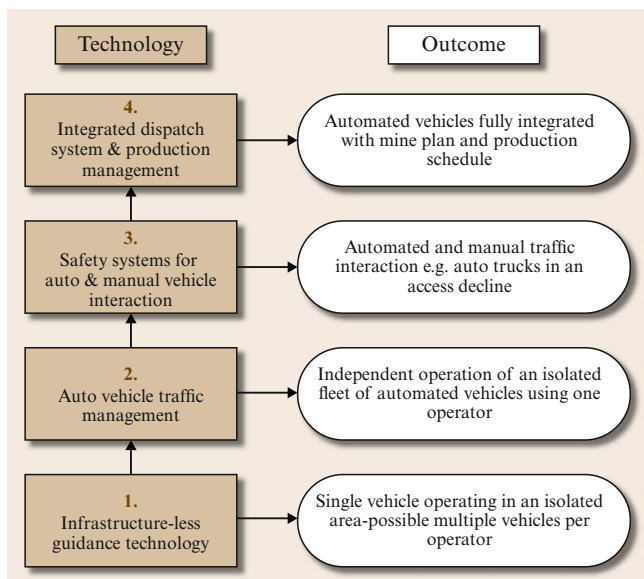


Fig. 49.9 The evolution of automated mining vehicle technology and the outcomes that could flow from each stage

be odometry and some other absolute system such as radiofrequency identification (RFID) tags to correct the drift of the odometry.

The availability of the two-dimensional (2-D) laser scanner in the mid to late 1990s saw the development of new-style LHD guidance systems. Some of these systems worked by using reflective markers placed along the mine walls at regular intervals (tens of metres) [49.16]. The LHD localized itself using a map of the beacons and the observed range and bearing readings to the beacons as observed by a 2-D laser scanner. This data was fused with odometry information and inertial data (such as heading rate and acceleration). Another system used the detected free space ahead of the vehicle, as determined by a 2-D laser scanner to reactively steer the vehicle [49.17]. This system also used the topology of the mine, again as determined by the 2-D laser scanner data, to localize the vehicle for high-level navigation purposes such as speed setting, turn decision making at intersections, etc. This approach was successfully demonstrated in a production mine in mid-1998, was subsequently commercialized and is now available as a product. Other approaches to feature-based localization underground are described in [49.18, 19]. Automated tramming systems for LHDs are available commercially from Tamrock, Caterpillar and Atlas Copco and market uptake is growing rapidly [49.1].

Traffic control is another important issue particularly in *mixed fleets* of human-driven and robotic vehicles. Currently each mine vehicle is equipped with VHF/UHF radio and local communication procedures are adopted to manage traffic. In some mines a decline is divided into segments or *blocks* and traffic lights (block lights) are used to indicate that a segment is occupied. A set of rules is used to give priority to ascending vehicles and heavy vehicles. A common control for autonomous vehicles is segregation of autonomous and manual equipment operations through the establishment of safety barriers to exclude people from the operational area. However these barriers are quite extensive and difficult to maintain, and cause significant operational constraints. Reliable collision and pedestrian detection is required.

Drilling

Semiautomated drilling of blast holes for the purpose of roadway/tunnel development or in open stope mining is now a reality. Drilling machines are available that can drill a ring of holes autonomously with little supervision over periods of a shift or more. There are strong control challenges with this class of machine due to the large outboard mass of the drill, the long telescoping

arm which is compliant, combined with flexibility in the hydraulic actuators and the mobile base.

Drills are still manually placed and setup, and autonomous mobility and positioning would be the next development.

Explosive Placement

The loading of drilled holes with explosives, detonators, and primers is still an active area of robotics research [49.20]. Current practices see miners assembling detonators and primers and explosives are either pumped or placed manually into holes (with the aid of hydraulic machinery). Detonators and wiring are installed and linked back to an *exploder box* to initiate the blast (sometimes from a central location). This job is hazardous, not because of the handling of explosives, but because of the confined nature of the operation and the proximity of the miners to the hydraulic equipment (booms) and the solid tunnel walls. There are a number of robotics challenges associated with this task including locating the drill holes, assembling the primers and detonators, and inserting the hose into the drill hole. The positioning problem is hard for the same reason as for drill control: a large mass to be positioned using a nonrigid manipulator. End-effector-mounted sensors can be used to map the holes and provide the fine control to guide the explosive charging tube into the hole. Explosives and associated detonators can then be pumped into the holes as required.

Secondary Rock Breaking

One of the aims of underground hard rock mining to break the rock (most commonly with explosives) to a size that can be easily handled by the haulage vehicles, conveyors, and lifts. The spacing of blasting holes, their length, and the quantity of explosives used are optimized to achieve the ideal fragmentation size of the rock. However, sometimes a boulder is created that is too large to be handled. These rocks are typically broken with additional explosives in situ. Particularly problematic are large rocks that jam the flow of ore in the draw point of a block cave mine. Drilling and blasting these rocks is very hazardous and would be an a strong candidate for robotics.

There is a class of smaller boulders that can often be moved by an LHD or haul truck but that is too large to be fed into a rock crusher before lifting to the surface. The chute to a rock crusher is guarded by a grill called a *grizzly*. The size of the grill openings is made so that only rocks small enough to be crushed may pass through. Larger rocks do not fit and must therefore be broken

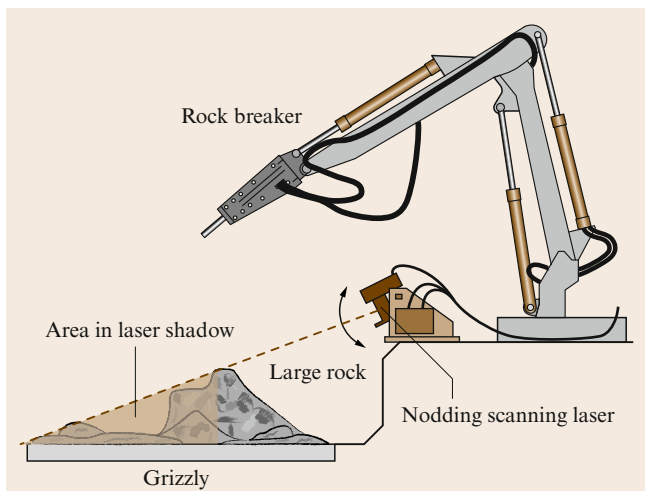


Fig. 49.11 Mechanical rock breaker

on the grizzly in order to pass through it. This job is carried out using a mechanical rock breaker, as shown schematically in Fig. 49.11. Rock breakers consist of a hydraulic or pneumatic jackhammer with a pick tool on the end mounted on a hydraulic multijointed arm driven by a human operator. The task of rock breaking is a logical one to be automated as, due to its infrequent nature, it is hard to justify a full-time operator when oversized rocks may occur a few times a day. The key research issues are sensing and perception, detecting when a rock needs to be broken, or simply needs nudging through the grizzly and then determining where on the rock to place the jackhammer pick in order to break the rock [49.21].

Metalliferous Roadway Development

Underground metalliferous mines use tunnels or roadways to access ore and facilities. These tunnels may sometimes be called declines, inclines, drifts, crosscuts, drives, and development headings depending on the function and region.

Traditional roadway development involves the steps shown in Fig. 49.12. Each cycle will extend the tunnel 3–6 m. The first step in the cycle is to survey and map the shape, location, and geology including grade and geological features. Note that the nature of the terrain has only been interpolated from drill core extracted at sparsely spaced locations. This work is done by surveyors and geologists before the ground has been supported. This is a hazardous zone which may not have settled since the previous blast. Research for robotic survey using photogrammetry, scanning lasers, and multispectral



Fig. 49.10 Tamrock drill jumbo for developing tunnels in metalliferous mines

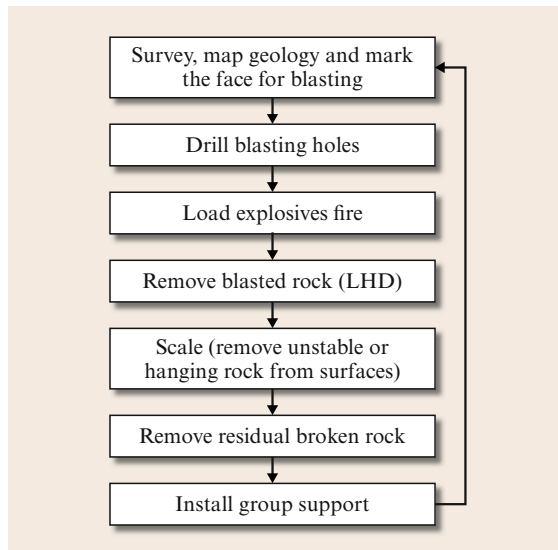


Fig. 49.12 Sequence of operations for tunnelling in metalliferous mines

analysis is in the early stages of development to acquire the necessary data without human entry.

Blast holes are drilled into the end of the drive in the direction of advance, typically using a drill jumbo (Fig. 49.10). Once blasted the rock is removed using an LHD to provide access for scaling. Scaling removes loose rock from the freshly blasted surface and this

is done either by hand, or using a drill jumbo. More recently successful trials have been undertaken using high-pressure water-jet scaling. Scaling is a hazardous task that is worthy of robotic control.

Ground support may be any combination of one or more of: rock bolts, steel mesh (typically 2×4 m sheets), sprayed reinforced concrete (shotcrete) sprayed plastic resin. It serves to increase the stability of the tunnel walls and roof, making the area safe for humans to work in. Methods of installation range from manual to mechanized using purpose-built machines. No automated methods are known at the time of writing, but development projects are underway. Ground support installation is particularly suited to robotics since the working environment is dangerous.

In some roadways where the rock is relatively soft or long straight sections are required, tunneling machines similar to those used in civil tunneling can be used, but this is uncommon. Tunneling machines use rock cutters which make slow progress in hard rock. However, rock cutting has some significant advantages and with the development of improved cutting technology this may become a common method of roadway development in the future. Rock cutting can produce smooth surfaces that are shaped for maximum stress tolerance and do not require the same level of preparation such as scaling or the installation of ground support.

Geology-Based Guidance Systems

As people are removed from the active mining zones, they can no longer use their senses to assess local conditions. Hence the introduction of specialized instrumentation is necessary to provide feedback. The use of scanning lasers and inertial navigation instruments is now accepted and is used for geometrical navigation and condition sensing. A further step change in guidance technology can be realized through the development and application of sensors that are sensitive to geological features including marker bands and mineralization. The use of such sensors is being trialled in coal mining, as mentioned in Sect. 49.3. In metalliferous mining, there are no current applications but significant opportunity exists, for example, hyperspectral imaging adapted from satellite imaging can be used to identify and map the mineralization of a face to help guide a mining machine. Fraser et al. [49.22] have shown that hyperspectral radiation is sensitive to minerals of interest and this could be developed to allow a metalliferous mining machine to follow ore seams.

49.2.3 Case Study: Nonconventional Robotic Miner

A proposed new mining method being developed [49.8] will deliver a safer and more economic way to mine a range of ore body types. It is a radical departure from conventional mining, exemplifying a new mining method which can only be achieved with robotic technology. Known as ROES™ it has been developed to solve many of the current mining occupational health and safety hazards, reduce mining costs, and reduce the time to bring ore into production. These benefits will also allow some existing subeconomic resources to be mined economically, thereby increasing the stocks of available ore (*reserves*).

The system involves vertical or inclined access to ore bodies rather than traditional horizontal access and will use robotic equipment for all activities associated with drilling, blasting and survey. This reduces the amount and size of tunneling required (*development*) compared with traditional methods. The level of control and real-time feedback to mine operators will also deliver greater mining coordination and flexibility. Figure 49.13 shows a comparison between the geometries of the ROES™ and sublevel open stoping mining methods.

ROES™ involves three main robotic activities: drilling, explosive placement, and survey. All of the

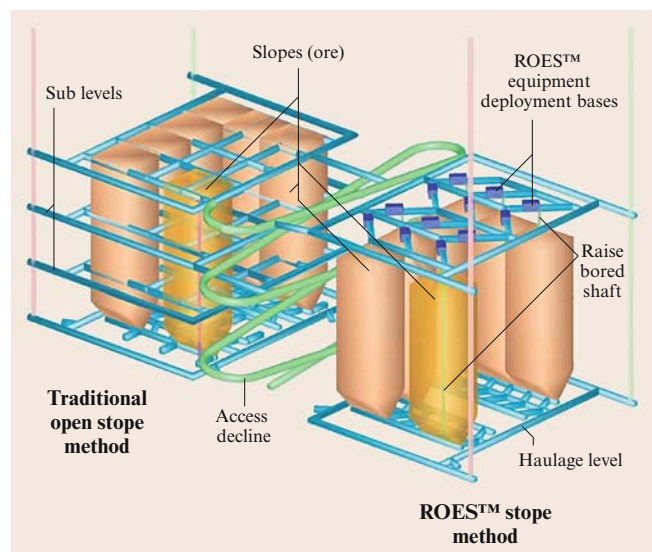


Fig. 49.13 Reduction in tunnelling required by ROES™ compared to traditional sub-level open stoping

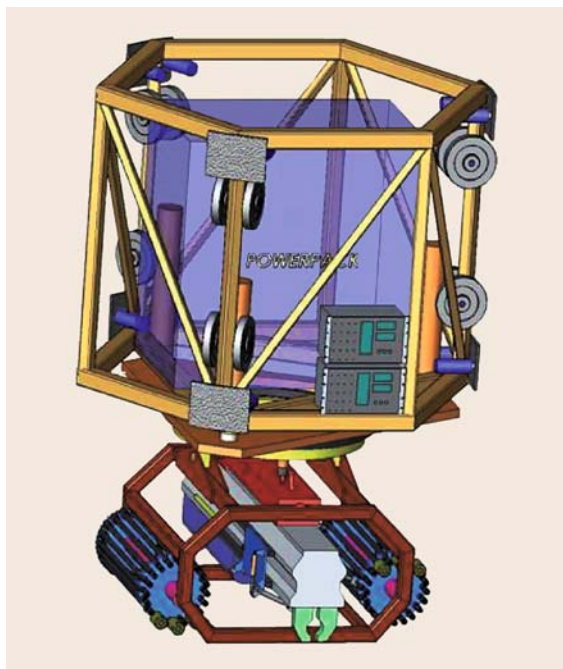


Fig. 49.14 Conceptual drilling module

equipment will be suspended from wire rope and deployed from the top of the vertical shaft to access the ore. The platforms will include onboard computers, PLCs, video cameras, machine encoders, and communications technology. Communication with the top of the shaft is possible via radio or infrastructure mounted into the

support and power cables. Such methods exist and are in use in the mining industry.

The drilling machine (Fig. 49.14) will be relatively small and stiff, which improves the drilling accuracy which is very important, so the pose and location of the drilling machine must be known in relation to the ore body and other structures. During travel to the drilling zone, one or a combination of several methods can be used to track the pose and location: odometry from the winder, high-level onboard inertial sensor, and laser distance measurement from the top of the shaft. The use of reference laser, beacons or novel methods that observe and track the movement of the machine against the wall of the shaft are also possible.

The rock drill is mounted separately from the main platform to provide control over its bearing, vertical position, and pitch. Once in the approximate drilling location, the main support platform can be wedged tightly in place against the shaft walls using hydraulic rams. The fine position of the drill can be controlled by a combination of electric and/or hydraulic actuators. Drill rods and spare drill bits are mounted on conventional carousels mounted either side of the drill and a robotic arm is used to transfer them to and from the drill.

The explosive placement platform will be similar in concept to the drilling platform. A survey platform will also be used to determine the condition of the shaft, drilled holes, stope void and size distribution of the blasted rock in the void below using sensors such as: millimeter-wave radar, photogrammetry and scanning lasers.

49.3 Underground Coal Mining

While many aspects of the coal extraction process, either by continuous miner or longwall-based methods are now highly automated, there has been a generally low rate of adoption of robotics in the underground coal mining industry worldwide. Conventional automated industrial process control systems are universally applied to fixed infrastructure coal conveying and processing elements of the mining chain, but there has been little commercial adoption of techniques to autonomously or tele-remotely control the operation of the basic mobile and semimobile coal extraction equipment. Commercial systems are currently limited to simple line-of-sight radio remote control of continuous miners and longwall shearers. In this section we will explain why the adoption of robotics has been slow in this environment.

The immediate underground coal mining environment is a hazardous area due to the presence of potentially explosive atmospheres. Accordingly, regulating authorities impose strict rules to govern the design and implementation of electrical equipment. This in turn translates to the necessity for high investment to develop robotics technology for a relatively limited market.

The coal face area is a dynamic environment with a high level of both expected and unexpected changes in geological conditions. This requires continuous observation of the current mining conditions and rapid and appropriate response to these occurrences to sustain the mining process. This function is carried out by experienced human operators. It is a challenging task to develop appropriate sensing methods both for local-

ization of the equipment and replacement of the human sensing ability to determine whether the equipment is mining coal or rock.

Despite these barriers, significant work has been done in developing robotics technology for this domain, particularly in navigation and sensing techniques and this section will explore significant issues, advances, and implementations.

49.3.1 Coal Mining Process

The nature of coal mining has led to the development of different styles of extraction equipment and potential robotics approaches. The longwall mining method, which accounts for the majority of underground worldwide production, is based on large equipment installations where motion of the individual components is highly repetitive and relatively constrained, whereas room (or bord) and pillar mining is more flexible, based on smaller mobile equipment.

Longwall Underground Coal Mining

In the longwall mining method ([49.23], Chap. 7) the primary extraction mechanism is a longwall shearer which has two ranging arms at the ends of which are rotating drums containing picks that remove the coal from the face. With reference to Fig. 49.15, the shearer travels along the face on a structure also housing an armored (or articulated) face conveyor (AFC) that carries the coal to one end of the face. The coal is then removed via other conveying systems to the surface. The roof is supported over this operation by a number of adjacent powered

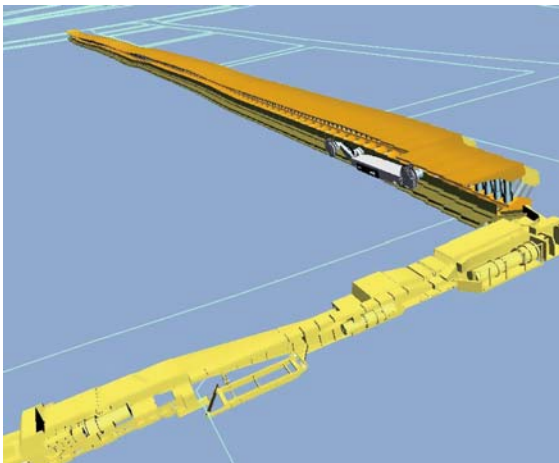


Fig. 49.15 Schematic of a longwall mining equipment installation

roof supports (also known as shields or chocks) which are each connected to the AFC. After the shearer passes, the chocks release from the roof, advance under hydraulic power into the cavity created by the removal of the coal, reposition the AFC and resupport the roof. The cycle then continues. Longwall face lengths range from 100 to 400 m and extraction rates can be of the order of 3000–4000 t/h. The advantages of the longwall method are the high sustained production rate and the ability to extract up to 80% of the coal seam.

Automation has been applied to various elements of the longwall mining process, particularly face alignment and cutting horizon control. Significant work has been done in the development of sensors to detect the interface between coal and rock in order to automatically keep the extraction process within the coal seam [49.24–27]. Face equipment manufacturers have produced automation systems that currently allow remote control of both face alignment and cutting horizon of longwall operations. The motion of powered roof supports can be initiated by the passage of the shearer. The sequence of support motion and the distance each hydraulic actuator moves are also programmable. Consequently a desired profile of the longwall face in plan view can be generated and input to the supports.

Room and Pillar Coal Mining

The alternative common underground coal mining method is room (or bord) and pillar ([49.23], Chap. 6) where the extraction system is based on the so-called continuous miner (Fig. 49.16). This is a much more mobile item of equipment than the longwall and can utilize many aspects of vehicle robotics technology. The boom raises and lowers the cutting drum and the unit is driven and steered through the track drives. The tail of the continuous miner then delivers cut coal to a flexible



Fig. 49.16 Continuous miner

conveying system or to shuttle cars ([49.28], Chap. 9) that in turn link to fixed conveyor belt infrastructure. The method requires less investment to establish than longwall and can operate in a wider range of geological conditions but has a lower production rate and does not extract as high a percentage of the available coal. All mainstream continuous miners are now operated via radio remote control so in a logical extension, robotics applications to this method include teleoperation, navigation, and control of roof and floor extraction horizons.

Operational Constraints

The underground coal mining environment presents unusual impediments to the implementation of robotics technology. Perhaps the most significant of these is the existence of explosive risk zones due to the presence of methane and coal dust in the atmosphere where mining equipment operates. There are various protocols for the implementation of equipment that could trigger an explosion. At one extreme, equipment can be designed so that energy levels are low enough to make the equipment unable to release sufficient energy to ignite explosive gases and thus be intrinsically safe. This greatly restricts the functionality of such equipment and the development costs are very high.

The other major implementation method is to house commercial equipment in purpose-built protective *flameproof* enclosures which can prevent an explosion inside the enclosure from propagating into hazardous external atmospheres. While this enables flexibility in the equipment that can be used, there are rules governing the design and testing of the enclosures, so development processes are slow and costly. There are problems implementing sensing systems requiring provision of transparent or at least dielectric windows in the enclosures for visible light, infrared, laser or radar-based sensors, and even computer screens for user interfaces. The dimensions of such windows are restricted in order to be explosion resistant. Moreover, such enclosures must not be opened during operation so live testing of operational hardware is virtually impossible. On the up side, the necessity to provide such robust housing for electrical and electronic equipment has a benefit in that protection is also afforded against the usual problems of water and dust.

Highwall Coal Mining

The above underground mining scenarios are currently largely manned operations. Highwall mining is a derivative of room and pillar mining and can only be



Fig. 49.17 A highwall mining operation

implemented by remote control. A series of parallel entries is made into the coal seam left exposed under the final highwall of a strip-mined open cut pit using a remotely controlled continuous miner. The coal is then conveyed to the surface by a continuous haulage method. In Fig. 49.17 continuous haulage is achieved using a series of interconnected conveyor cars. A *launch vehicle* is used to insert and remove cars, redirect the coal flow to a stockpile and as a location for an operator station. Supporting pillars of 1–3 m in width remain between adjacent entries, each of which can be up to 500 m long. If the entry spacing and pillar widths are not maintained then loss of equipment through roof falls and, in the extreme, catastrophic failure of a number of supporting pillars can result.

The rationale behind this method is that it is a low-cost way of extracting coal when overburden depths mean that open cut extraction methods are uneconomic. The method can extract about 60–70% of the available resource and then makes inaccessible deeper coal from other mining methods. Consequently the method is now falling from favor, being replaced by longwall operations with access roadways being driven directly into the seam from the highwall.

49.3.2 Scope for Robotics

Localization

Several of the methods described above require precise knowledge of the position of the cutting tool, and the underground environment is particularly challenging for accurate localization. To address this problem in the longwall context a shearer position measurement system (SPMS) based on inertial navigation has been developed [49.29] to accu-

rately measure the three-dimensional position of the shearer.

Inertial navigation (INS) technology is a logical method to measure machine position in the mining environment. State-of-the-art ring laser or fibre optic sensors are robust, often being designed for military applications. They require little or no connection to external sensors and thus can be located within the machine envelope in protected locations. The United States Bureau of Mines conducted tests using the Honeywell modular azimuth positioning system (MAPS) and subsequently the Honeywell ore recovery and tunneling aid (HORTA) units in the late 1980s for continuous miner guidance applications [49.30]. This work resulted in field trials in 1997 [49.31]. Further developments by CSIRO in Australia, resulted in the development of the first full-survey highwall mining guidance system in the mid-1990s [49.32, 33]. This system is now a critical requirement for highwall mining in Australia and is receiving increasing international acceptance.

The INS sensor is also applicable to longwall mining. If the real-time position and orientation of the shearer are continuously measured, the fact that the shearer passes over all the supports in a fixed geometrical relationship can be used to calculate the real-time positions of all the supports and thus the longwall face as a whole. This development, when added to existing manufacturers' capability, has facilitated reliable longwall face automation.

Planning and Control

Given better information about the position of the cutter and the position of the coal seam from geological models and online measurements it is possible to apply robotic path-planning techniques. However, true closed-loop control of face alignment has not been possible because there has been no reliable method to measure the true plan position of all the powered supports to provide accurate feedback to the control system. This process lends itself to the application of robotics in controlling the motion of the supports to manage face alignment and also the motion of the ranging arms on the shearer to position the extraction roof and floor horizons optimally in the seam.

Shuttle Cars

Currently the shuttle car operator docks the machine to the discharge conveyor of the continuous miner, controls the filling process, drives the shuttle car to a discharge point at a fixed conveyor, controls the emptying process, and returns to the miner to repeat the cycle. Often two

shuttle cars are used in parallel to increase throughput so there are scheduling issues when one unit is required to wait for the other to unload or load. The entire operation could be automated using navigation developments already applied to LHD vehicles in metalliferous mining as described in Sect. 49.2.2. However no prototype or commercial shuttle car automation systems have yet been reported.

49.3.3 Rapid Roadway Development

Often longwall mining operations are delayed because the necessary gate roads have not been completed. Ground support in these roadways is often provided by roof and rib (wall) bolting. This is a relatively involved process in which firstly a hole is drilled into the strata using a portable or machine-mounted drill rig. The drill steel is then withdrawn and bladders of epoxy are manually inserted into the hole. Finally the bolt is inserted into the hole, again using the drill rig. The bolt is spun to break the bladders which mixes the epoxy and hardener to bond the bolt to the strata. Often the bolt is then tensioned to compress the strata to form structural support of the roof and rib.

Placement of bolts has been identified as a limiting factor in the speed of roadway development so there has been considerable industry attention given to improvement of the speed of bolting. Unfortunately the conventional bolting procedure poses several automation challenges shared also with other underground drilling applications discussed in this chapter. Automatic relocation of the hole in order to insert different devices to inject the epoxy grout and then to place the bolt is difficult to achieve in the operating environment. The development of a self-drilling bolt has eliminated several of the automation issues because it is now possible to drill a hole with the bolt itself and then leave the

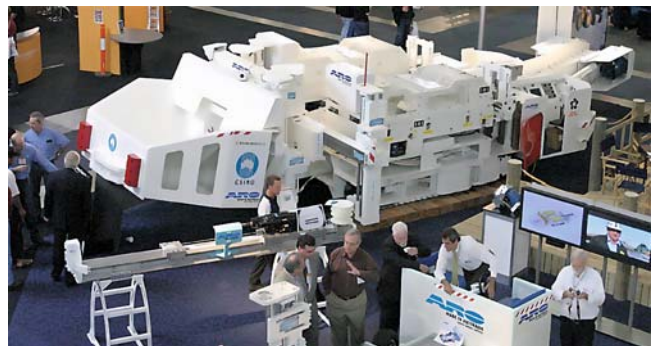


Fig. 49.18 Prototype machine for roof and rib bolting

bolt in place while the epoxy is introduced, mixed, and cured.

A system has been developed to implement automatic roof and rib bolting using a self-drilling bolt

concept [49.34]. The major elements of this system, shown in Fig. 49.18, are removable cassette storage of bolts, an automated bolt delivery system, and automatic drilling/bolting rigs

49.4 Surface Coal Mining

Many different minerals are extracted in surface mining operations, for example coal (Fig. 49.3), oil shales, bauxite, iron ore, and phosphate. We can consider two different excavation requirements: removing the overburden which often covers the commodity of interest, and the commodity itself. The remainder of this section will discuss the general principles of excavation and material haulage using open-pit coal mining as an example. Most aspects of the process, and the types of equipment required, are common to many other open-pit mineral extraction operations.

A consequence of many years of commodity extraction is that all the remaining commodity is buried deeper than what has been removed to date. Miners refer to a stripping ratio, which is the ratio of the overburden volume that must be removed to access one tonne of mineral. Over the last 20 years, in the Australian coal industry for example, this ratio has climbed from 2–5:1 to 8–15:1 today. More material must be moved to reach the mineral, requiring a massive improvement in the productivity of overburden removal and a reduction in cost in order to maintain profitability – this is the challenge that automation and robotics can contribute to.

Various types of machine are commonly used for excavation. In approximately increasing size order: front-end (or wheel) loaders, back hoes, bulldozers, rope shovels, hydraulic excavators, draglines, and bucket wheel excavators. Front-end loaders, back hoes, and bulldozers are mobile plant that can be rapidly deployed as required. Their disadvantage is a relatively low productivity in dirt movement. Draglines and bucket wheel excavators have high capital costs, but operating cost per volume moved is very low.

From a robotic perspective, there has been important prior work in automating excavation machinery. *Singh* [49.28] provides a good treatment of much of this work using a number of implemented systems to illustrate the state of the art. Work at Carnegie Mellon University by *Singh*, Cannon, *Rowe*, *Stenz* and others in the 1990s, culminated in a comprehensive study and experimental evaluation of an autonomous 25 t hydraulic backhoe-type excavator [49.35]. This work in-

cluded the development of techniques to estimate soil hardness and dig forces which were integrated into an autonomous digging control law, as well as methods for planning dig locations and clean-up operations. Other work in this area can be found in [49.36–40]. The smaller machines are typically hydraulically actuated and the control issues associated with such machines has been investigated by *Tafazoli* et al. [49.41] and *Bonchis* et al. [49.42,43]. Rope-shovels and hydraulic excavators are discussed in Sect. 49.4.2.

The remainder of this section will discuss the two main approaches to overburden removal for coal mining: the dragline excavator, and the truck-shovel operation in which a rope shovel or hydraulic excavator loads material into a truck.

49.4.1 Dragline Excavator

The dragline (Fig. 49.19) is the tool of choice for overburden stripping. Invented in 1904 for use digging the Chicago canal, the fundamental principles have changed little since, but they have evolved into truly massive electrically powered machines; see Fig. 49.3. Modern machines typically have boom lengths of over 100 m, weigh up to 5000 t, with bucket capacities of the order of 100 t.

They comprise a rotating assembly that includes the *house* (drive motors, controls, and operator cabin), *tri-structure* or *mast*, and *boom*. The house rotates on a bearing surface on top of the *tub* which sits on the ground. A large-diameter ring gear is fixed to the tub and the house is rotated by a number of pinions driven by motors in the house. While earlier and smaller draglines used a tracked base, the modern walking dragline drags its tub along the ground by means of large eccentrically driven *walking shoes* at the side of the machine. The dragline has three driven mechanical degrees of freedom: the house and boom rotate with respect to the tub; the bucket can be hoisted by a cable passing over sheaves at the tip of the boom; and the bucket can be dragged toward the house by a cable passing over sheaves at the base of the boom (Fig. 49.19). During digging, the

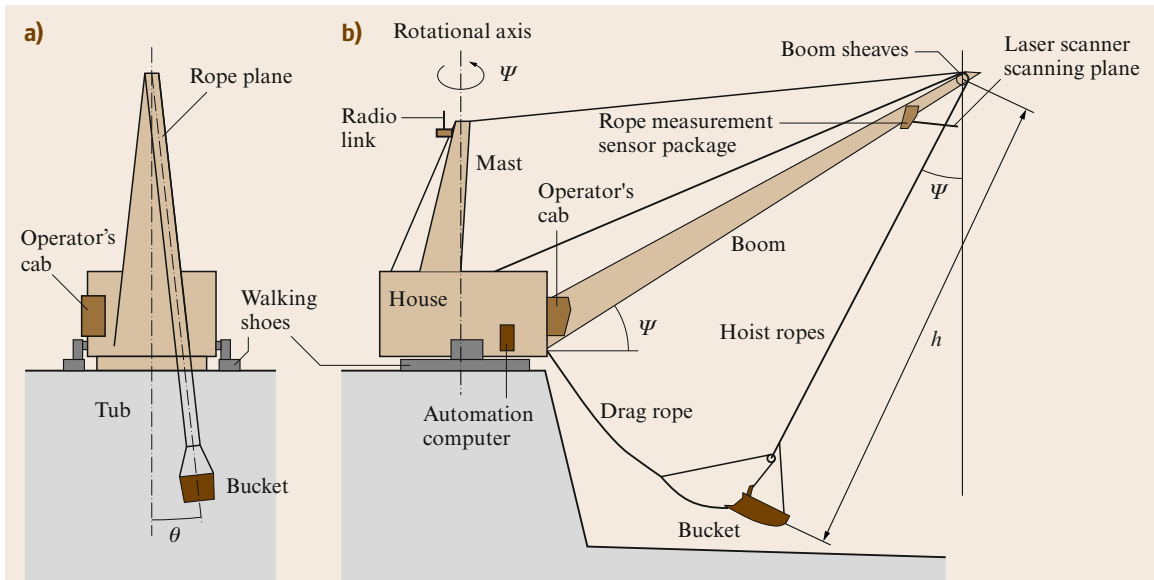


Fig. 49.19 Schematic of dragline

bucket motion is controlled using only the drag and hoist ropes. When the bucket is filled it is hoisted clear of the ground and swung to the dump position by slewing the house and boom. The drag and hoist drives now control the position of the bucket within a vertical plane that contains the centreline of the boom however the bucket is free to swing normal to that plane. Interestingly the bucket has three degrees of freedom within the boom plane but only two control inputs, making control of bucket orientation a complex function of position, speed, and rope tension [49.44].

Capital cost of these machines is of the order \$US 40–80 M and the operating cost is approximately \$US 5 M per year (labor, power, and maintenance) working 24/7. A dragline can only excavate to 50 m and with pit depths now well over 100 m the overburden must be removed in two dragline passes, or by using shovels and trucks (Sect. 49.4.2) to prestrip. The threshold depth at which underground mining methods would be employed continues to increase.

The issues facing owners of these machines are: maximizing productivity (m^3/shift) in the presence of operator variability and maintenance costs (around 30% of operating cost). It has been estimated that a 1% improvement in machine productivity is valued at around \$US 1 M per machine per year. Operation is cyclic, and each cycle moves up to 100 t of overburden, and typically takes 60 s of which 80% of the time is swinging the bucket through free space. The work is repetitive,

yet requires considerable skill. Performance can vary by up to 20% across the operator population, and for an individual will vary over a shift (up to 12 h).

The solutions currently pursued include: operator performance monitoring, improved training, and routine overloading of the machine which exploits the generous design safety margins and trades off productivity and maintenance costs.

Robotics offers the possibility of control that minimizes cycle time while bounding the stresses that cause machine damage. The challenges are to replace the human function of visually sensing the state of the bucket, its rigging and the local terrain and actuate the drag, hoist (joystick) and slew (pedal) controls in a coordinated fashion, controlling and exploiting the bucket's natural tendency to swing, while working within the significant voltage, current, and power constraints of the motors.

Early attempts at dragline automation [49.45] simply replayed the operator's control inputs but this failed because the control inputs are based on the instantaneous machine state which is a history of the initial condition and subsequent inputs, as well as the initial and goal states which vary from dig to dig. A key challenge is sensing the machine and bucket state which would include encoders on the motors and a noncontact sensor of bucket angles since the bucket is treated very roughly, being pulled through several meters of broken rock. Noncontact solutions involving computer vision [49.46]

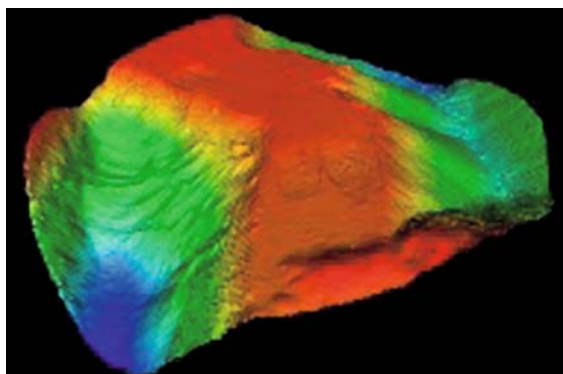


Fig. 49.20 Terrain mapping from the tip of a dragline boom

and laser range finders [49.47] have been proposed. Once equipped with sensors a dragline can be considered as a three-DOF (degree of freedom) robot (drag rope, hoist rope, slew angle) with a flexible, or unactuated, final link [49.48]. There are two control problems. Firstly the tool-tip (bucket) motion along a defined trajectory in free space [49.49] which is similar to that of controlling cable suspended loads for cranes [49.50], and consumes 80% of the cycle time. In a major production trial [49.51] a dragline automation system performing this free-space motion moved over 250 000 t of overburden in 3042 cycles over a 2 week period. Performance was equivalent to a human operator but the automation system had to lift the bucket higher in order to compensate for uncertainty in the environment. Mapping the surrounding terrain [49.52], as shown in Fig. 49.20, would allow the automation system to plan time-optimal collision-free paths.

The second control problem is moving the bucket through the ground in order to fill it, and determining when it is full. Some initial work in this area is reported in [49.53].

Another important consideration is a human interface that allows fast and seamless transfer between automatic and human-controlled operation [49.49].

49.4.2 Truck–Shovel Operation

As implied by the name the truck–shovel operation has two different machine types: a rope shovel (Fig. 49.21) or hydraulic excavator (Fig. 49.22) to excavate material and load it into the truck which hauls it away. While draglines have the lowest cost per unit volume removed (at least ten times more cost effective than a truck and shovel operation) they are very significant capital assets and difficult to move quickly around the mine. By



Fig. 49.21 A rope shovel dumping into a haul truck. A truck typically holds between one and three dipper loads of material (©CSIRO)



Fig. 49.22 A hydraulic excavator (©Automated Positioning System, Brisbane)

contrast trucks and shovels can form more *flexible* operations since smaller individual units can be added or subtracted or readily deployed around the mine site.

Truck–shovel combinations can be used for overburden removal, either totally or in a prestripping mode in preparation for a dragline. Trucks and shovels are very commonly used to excavate the exposed coal seam.

The modern shovel (Fig. 49.21) is descended from the 19th century steam shovel and employs large electric drives and a cable transmission. Hydraulic excavators have a dipper mounted on the end of a rigid five-DOF hydraulic arm, typically on a tracked base. These machines are kinematically much closer to industrial robots, but on a massive scale, with dipper sizes over 20 m³ available.

From an automation perspective [49.53] many of the issues are the same as for draglines: filling the bucket with broken rock, planning a time-optimal collision-

free trajectory, and human interface. The rigidity of these machines simplifies the estimation of bucket tooth position.

Typically these machines swing and dump material into a haul truck, see Fig. 49.21. A perception system could be used to detect the truck and dump the material into a free area of the tray. Planning the motion of the trucks and the shovel becomes a problem in multirobot scheduling and path planning.

Blasting

The overburden is typically a soft sandstone, and the coal seam itself, after millions of years of compression, also resembles rock. In order to excavate it must first be blasted to fracture the material into pieces that can be excavated. A matrix of blast holes is drilled in the overburden layer and fired prior to overburden removal. The process is repeated with the exposed coal seam.

This process is amenable to automation, though it has not yet been demonstrated. The drill rig is a large-wheeled machine which needs to be positioned on the bench, and the drilling process itself needs to be automated. The drill string is assembled from short components as the drilling progresses. Online feedback from the drill head, speed, and torque, can provide geological data about the strata being drilled through. Accurate knowledge of the drill head is also required, since it can be deflected by geological conditions causing the blast to deviate from the design. Once the holes are all drilled there is scope for automating the explosive loading process: currently large tankers of explosive that visit each hole, insert a hose into the hole along with primers and detonators, and pump the explosive.

49.4.3 Surface Haulage

In surface mines, material (commodity or overburden) must be moved from where it is broken to where it will be processed or stockpiled. Haul trucks are the preferred means of moving material around a surface mine, typically diesel-electric vehicles with payloads of up to 400 t. The automation of these haul trucks has been an ongoing research and development activity for the truck manufacturers for over 10 years. Both Caterpillar and Komatsu (the two largest manufacturers of haul trucks) have had very active automation research programs. The key robotics challenges to the automation problem are

localization and obstacle detection. GPS is an obvious choice as a primary localization sensor. However, GPS is not reliable close to the highwalls of a typical open pit (due to a lack of observable satellites and due to multipath issues). The ramps out of pits typically hug the walls, which makes the use of GPS as a localization sensor difficult. This problem can be solved using pseudo-lites to cover the GPS dead spots or other sensors such as laser scanner, radar, inertial guidance, and odometry in these areas. Both Caterpillar and Komatsu demonstrated autonomous driving of haul trucks in the late 1990s [49.54].

It is the area of obstacle detection that is far more challenging. Autonomous haul trucks must be capable of detecting rocks on the road in front that have fallen from other trucks. They may be required to detect other vehicles (if the haul road is open to other mine traffic) and they may be required to detect mine personnel. The development of robust obstacle detection systems for these vehicles is difficult because of the relatively harsh conditions encountered in mining environments. The operating environment could include rain, dust, mud, high humidity, diesel fumes (small particles), extremes of temperature, severe vibration, extreme vehicle pitching and rolling, and bright light sources (e.g., the sun). [49.55] shows how a 2-D scanning laser can be used to map the terrain ahead of a mining vehicle and hence be used to detect obstacles. Scanning lasers have been widely used for obstacle detection in the DARPA Grand Challenge [49.56] where nearly all competitors used such sensors as the primary means of detecting obstacles.

The technology for autonomous trucks has now been established but there are still very few examples of mines using autonomous haul trucks in regular production. It seems that the technology is ahead of the culture in this case and the mining industry is not quite ready for this big leap in autonomous operation. Instead examples of semiautomation or driver assists are starting to be tried. An example of this is the work in [49.57] where a collision warning system for haul truck drivers has been developed. This system uses many of the sensing systems of an autonomous truck but instead of directly controlling the truck the driver is alerted when he/she are starting to head off the road (maybe due to tiredness or inattention), or when another truck is on a potential collision course.

49.5 Conclusions and Further Reading

Today, mining remains hard and hazardous work with a wide scope for the application of robotics. The main drivers are the need for greater productivity and improved health and safety for mine personnel. Mining robots differ considerably from traditional factory robots in that the mining environment is ever changing, that is, the purpose of mining, and the machines are typically hydraulically actuated, and are often diesel powered. These challenges have meant that robotics has not been rapidly adopted by the mining industry to date, but it is clear that the mining industry will have to adopt more automation, and with it robotics, to further significantly improve productivity and safety as traditional methods such as staff training, improved work practices, and larger and improved machine design are providing diminishing returns.

Further reading on the state-of-the-art robotic mining machines in operating mines as at 2006 can be obtained from [49.1] and examples of current robotic miners include

- automated mobile haulage trucks in underground metal mines
- automated drilling machines [49.58, 59]

Areas of development, currently being researched, include

- a large and early challenge will be safe operation of mixed fleets, where robotic and manned vehicles operate in the same workspace,
- teleoperation, with immersive environments, is likely to grow in usage, but with the operators located thousands of kilometers away instead of actually at the mine site as is normal today. These telemining technologies will be essential when it comes to mining the rich but difficult to access mineral resources deep under the ocean [49.60] and on planets and asteroids,
- in situ processing of mineral ore by leaching to extract the metal, or in situ gasification of underground coal, remove the need to excavate and transport the material, much of which is waste. These methods will all require underground operations that are suited to robotics [49.61],
- microtunneling machines [49.62], or moles, will follow arbitrary paths through the ground, releasing gas from coal seams, or following narrow bands of commodity. Research in propulsion, localization, and communications will be required to make this a reality,
- automated explosive placement [49.20] and stoping methods [49.8],
- automated tunneling including installation of ground support [49.34],
- robotic rock cutting machines [49.63].

References

- 49.1 P. Moore: Return of the robominers, *Mining Mag.* **195**(3), 16–22 (2006)
- 49.2 Y. Potvin, P. Nedin, M. Sandy, K. Rosengren, M. Rosengren: *Towards the Elimination of Rockfall Fatalities in Australian Mines* (Australian Centre for Geomechanics, Crawley 2001), Tech. Rep.
- 49.3 H.L. Hartman (Ed.): *SME Mining Engineering Handbook*, Vol. 1,2, 2nd edn. (Society for Mining, Metallurgy and Exploration, Shafter 1992)
- 49.4 J.T. Woodcock, J.K. Hamilton (eds.): *Australasian Mining and Metallurgy: The Sir Maurice Mawby memorial*, Vol. 2 (The Australasian Institute of Mining and Metallurgy, Alabama 1993)
- 49.5 W.A. Hustrulid, R.L. Bullock (Eds.): *Underground Mining Methods* (Society for Mining, Metallurgy and Exploration, Shafter 2001)
- 49.6 S. Belfield: Fmg aims for a cut above the rest, *Austral. Mining* **99**(5), 22 (2007)
- 49.7 J. Raimoaho: Mine-wide communication system, *Proceedings of the 4th Regional Symposium on Computer Applications in the Minerals Industries* (Tampere 2001) pp. 181–189
- 49.8 J. Cunningham, I. Gipps: Practical & strategic technology choices for advanced mining, *Proceedings of the 12th International Symposium on Mine Planning and Equipment Selection* (The Australasian Institute of Mining and Metallurgy, Kalgoorlie 2003) pp. 19–27
- 49.9 E. Villaescusa, L.B. Neindorf, J.B. Cunningham: Bench stoping of lead/zinc orebodies at mount isa mines limited, *Proc. Joint Japan Australian Inst. Mining Metallurgy Symp.* (The Australasian Institute of Mining and Metallurgy, Uke 1994)
- 49.10 G. Eriksson, A. Kitok: Automatic loading and dumping using vehicle guidance in a Swedish mine, *Int. Symp. Mine Mechan. Autom.* (Colorado 1991) pp. 15.33–15.40
- 49.11 N. Vagenas, H. Sjöberg, S. Wokström: Application of remote-controlled/automatic Load-Haul-Dump system in Zinkgruvan, Sweden, *Int. Symp.*

- Mine Mechan. Autom. (Colorado 1991) pp.6.21–6.30
- 49.12 R. Hurteau, M. St-Amant, Y. Laperriere: Optically guided LHD: A demonstration prototype, Int. Symp. Mine Mechan. Autom. (Colorado 1991) pp.6.11–6.20
- 49.13 A. Grenier, G. Chevette, C. Coache: Noranda automatic guidance system architecture overview, Canadian Symp. Mine Autom. (Montreal 1994) pp.116–119
- 49.14 G. Brophy, D. Euler: The Opti-Trak System, a system for automating today's LHDs and trucks, Int. Symp. Mine Mechan. Autom. (Sweden 1993) pp.403–409
- 49.15 G. Brophy, D. Euler: The Opti-Trak System, a system for automating today's LHDs and trucks, CIM Bull. **87**(984), 52–57 (1994)
- 49.16 S. Scheduling, E. Nebot, M. Stevens, H. Durrant-Whyte, J. Roberts, P. Corke, J. Cunningham, B. Cook: Experiments in autonomous underground guidance, Proc. IEEE Int. Conf. Robot. Autom. (Albuquerque 1997) pp.1898–1903
- 49.17 J. Roberts, E. Duff, P. Corke: Reactive navigation and opportunistic localization for autonomous underground mining vehicles, Int. J. Inform. Sci. **145**, 127–146 (2002)
- 49.18 D. Silver, J. Carsten, S. Thayer: Topological global localization for subterranean voids, Field and Service Robotics: Results 5th Int. Conf., Vol. 25, ed. by P. Corke, S. Sukkariah (Springer, 2006) pp.117–128
- 49.19 J. Larsson, M. Broxvall, A. Saffiotti: A navigation system for automated loaders in underground mines, Field and Service Robotics: Results 5th Int. Conf., Vol. 25, ed. by P. Corke, S. Sukkariah (Springer, 2006) pp.355–366
- 49.20 A. Young: The ELAP project, Engineering Dimensions **23**(2), 20–21 (2002)
- 49.21 P. Corke, J. Roberts, G. Winstanley: 3D perception for mining robotics, Proc. Int. Conf. Field Service Robot. (Canberra 1997) pp.41–47
- 49.22 J. Raimoaho, S.J. Fraser, L. Whitbourn, K. Yang, E. Ramanaidou, P. Connor, G. Poropat, P. Soole, P. Mason, D. Coward, R. Phillips: Mineralogical face-mapping using hyperspectral scanning for mine mapping and control, Proc. Sixth Int. Mining Geology Conf., Rising to the Challenge, ed. by S. Dominy (The Australasian Institute of Mining and Metallurgy, Darwin 2006) pp.227–232
- 49.23 R.D. Singh: *Principles and Practices of Modern Coal Mining* (New Age International, New Delhi 1997)
- 49.24 S.L. Bessinger, M.G. Nelson: Remnant roof coal thickness measurement with passive gamma ray instruments in coal mines, IEEE Trans. Ind. Appl. **29**, 562–565 (1993)
- 49.25 G.L. Mowrey: Promising coal interface detection methods, Mining Eng. **43**(1), 134–138 (1991)
- 49.26 R.L. Chufo, W.I. Johndon: A radar coal thickness sensor. In: *New technology in mine health and safety*, ed. by A.W. Khair (SME, Phoenix 1992) pp.337–347
- 49.27 G. Stolarczyk, G.L. Stolarczyk, K.L. Perry: Horizon sensor for advanced coal extraction (ace), National Mining Association, MINExpo (Las Vegas 1996)
- 49.28 S. Singh: The state of the art in automation of earthmoving, ASCE J. Aerosp. Eng. **10**(4), 179–188 (1997)
- 49.29 M.S. Kelly, D. Hainsworth, D. Reid, P. Lever, H. Gurgenci: Longwall automation: A new approach, Third International Symposium on High-performance Mine Production Longwall, Stoping, other Mining Methods (Aachen 2003) pp.117–132
- 49.30 J.J. Sammarco: Field evaluation of the modular azimuth and positioning system (maps) for a continuous mining machine, Tech. Rep. IC 9354 (Bureau of Mines, U.S. Department of the Interior 1993)
- 49.31 W.H. Schiffbauer: Pursuit of accurate navigation and control of continuous mining machines for coal mining, Proc. 4th Int. Symp. Mine Mech. Autom. (Brisbane 1997) pp.B6.33–B6.46
- 49.32 D. Reid, D.W. Hainsworth, R.J. McPhee: Lateral guidance of highwall mining machinery using inertial navigation, Int. Symp. Mine Mechan. Autom. (Brisbane 1997)
- 49.33 D.C. Reid, J.C. Ralston, D.W. Hainsworth, R.J. McPhee, E.K. Matejowsky: Highwall mining guidance: A major advance in highwall mining teleoperation, Int. Adv. Robot. Program Workshop (IARP 2000) (Brisbane 2000)
- 49.34 D.W. Hainsworth, D.C. Reid, R. McPhee, J.C. Ralston, P. Corke, G. Winstanley: Automation applied to a rapid roadway development system for underground coal mining, Proc. 4th Reg. Symp. Comp. Appl. Miner. Ind. (Tampere 2001) pp.23–31
- 49.35 A.T. Stentz, J. Bares, S. Singh, P. Rowe: A robotic excavator for autonomous truck loading, Auton. Robot **7**, 175–186 (1999)
- 49.36 D.A. Bradley, D.W. Seward: The development, control and operation of an autonomous robotic excavator, J. Intell. Robot. Syst. **21**(1), 73–97 (1998)
- 49.37 L.E. Bernold: Principles of control for robotic excavation, Eng. construct. oper. in space III: Space '92; Proc. 3rd Int. Conf., Vol. 2 (A93–41976 17–12) (Denver 1992) pp.1401–1412
- 49.38 Q. Ha, M. Santos, Q. Nguyen, D. Rye, H. Durrant-Whyte: Robotic excavation in construction automation, Robot. Autom. Mag. **9**, 20–28 (2002)
- 49.39 A. Hemami: A fundamental analysis of robotic excavation, J. Aerosp. Eng. **8**(4), 175–179 (1995)
- 49.40 S. Sarata, W. Yossawee, T. Tsubouchi: Approach path generation to scooping position for wheel loader, Proc. Int. Conf. Robot. Autom. (2005) pp.1809–1814
- 49.41 S. Tafazoli, P.D. Lawrence, S.E. Salcudean: Identification of inertial and friction parameters for

- excavator arms, IEEE Trans. Robot. Autom. **15**, 966–971 (1999)
- 49.42 A. Bonchis, P.I. Corke, D.C. Rye: Experimental evaluation of position control methods for hydraulic servo systems, IEEE Trans. Control Systems Technol. **10**, 876–882 (2002)
- 49.43 A. Bonchis, P. Corke, D. Rye: A pressure-based, velocity independent, friction model for asymmetric hydraulic cylinders, Proc. IEEE Int. Conf. Robot. Autom. (Detroit 1999) pp. 1746–1751
- 49.44 P. Ridley, P. Corke: Calculation of dragline bucket pose under gravity loading, Mech. Mach. Theory **35**, 1431–1444 (2000)
- 49.45 L. Allison: Creative engineering – a key to the future of the Australian coal industry, Proc. Conf. Inst. Mining Electr. Mining Mech. Eng. (Newcastle 1992)
- 49.46 D.W. Hainsworth, P.I. Corke, G.J. Winstanley: Location of a dragline bucket in space using machine vision techniques, Proc. Int. Conf. Acoust. Speech Signal Process. (ICASSP-94), Vol. 6 (Adelaide 1994) pp. 161–164
- 49.47 J. Roberts, F. Pennerath, P. Corke, G. Winstanley: Robust sensing for a 3500 t field robot, Proc. IEEE Int. Conf. Robot. Autom. (Detroit 1999) pp. 2723–2728
- 49.48 J. Roberts, P. Corke, G. Winstanley: Development of a 3500 t field robot, Int. J. Robot. Res. **18**, 739–752 (1999)
- 49.49 P.I. Corke, J.M. Roberts, G.J. Winstanley: Experiments and experiences in developing a mining robot system. In: *Experimental Robotics VI*, Lecture Notes in Control and Information Sciences, Vol. 232, ed. by P. Corke, J. Trevelyan (Springer, Sydney 2000) pp. 183–192
- 49.50 A. Ridout: Variably damped swing contro, of the overhead crane, Proc. 15th Annual Conf. IEEE Ind. Electr. Soc. (1989) pp. 263–269
- 49.51 P. Corke, G. Winstanley, M. Dunbabin, J. Roberts: Dragline automation: Experimental evaluation through productivity trial, Proc. Int. Conf. Field Serv. Robot. (Lake Yamanaka 2003) pp. 249–254
- 49.52 J. Roberts, G. Winstanley, P. Corke: 3D imaging for a very large excavator, Int. J. Robot. Res. **22**, 467–478 (2003)
- 49.53 M. Dunbabin, P. Corke: Autonomous excavation using a rope shovel, J. Field Robot. **23**, 379–394 (2006)
- 49.54 J. Chadwick: Autonomous mine truck, Mining Mag. **175**(5), 287–288 (1996)
- 49.55 J. Roberts, P. Corke: Obstacle detection for a mining vehicle using a 2d laser, Proc. Austral. Conf. Robot. Autom. (Melbourne 2000) pp. 185–190
- 49.56 K. Iagnemma, M. Buehler: Special issue on the DARPA grand challenge (part 1), J. Field Robot. **23**(8), 461–462 (2006)
- 49.57 E. Nebot, J. Guivant, S. Worrall: Haul truck alignment monitoring and operator warning system, J. Field Robot. **23**(2), 141–161 (2006)
- 49.58 C.L. Jimeno, E.L. Jimeno, F.J.A. Carcedo: *Drilling and Blasting of Rocks* (A.A. Balkema, Rotterdam 1995)
- 49.59 Committee on Advanced Drilling Technologies, Drilling and Excavation Technologies for the Future. (National Research Council 1994)
- 49.60 C. Yeats, T. McConachy: Deep blue minerals: Towards a sustainable marine minerals industry, Tech. Rep. P2005/135 (CSIRO Exploration and Mining, 2005)
- 49.61 R.W. Bartlett: *Solution mining: leaching and fluid recovery of materials* (Gordon and Breach Science, Philadelphia 1992)
- 49.62 M. Wendt, G. Einicke: Development of a water-hydraulic self-propelled robotic drill for underground mining, Field and Service Robotics: Results 5th Int. Conf., Vol. 25, ed. by P. Corke, S. Sukkariah (Springer, 2006) pp. 355–366
- 49.63 F. Delabbio: Hardrock excavation alternatives – present status and future options?, WRBA 2003 Conference on Expanding the limits of mechanical excavation (2003)

Search and

50. Search and Rescue Robotics

Robin R. Murphy, Satoshi Tadokoro, Daniele Nardi, Adam Jacoff, Paolo Fiorini, Howie Choset, Aydan M. Erkmen

In order to summarize the status of rescue robotics, this chapter will cover the basic characteristics of disasters and their impact on robotic design, describe the robots actually used in disasters to date, promising robot designs (e.g., snakes, legged locomotion) and concepts (e.g., robot teams or swarms, sensor networks), methods of evaluation in benchmarks for rescue robotics, and conclude with a discussion of the fundamental problems and open issues facing rescue robotics, and their evolution from an interesting idea to widespread adoption. The Chapter will concentrate on the rescue phase, not recovery, with the understanding that capabilities for rescue can be applied to, and extended for, the recovery phase. The use of robots in the prevention and preparedness phases of disaster management are outside the scope of this chapter.

50.1 Overview	1152
50.1.1 Motivation	1152
50.1.2 Rescue Robot Tasks	1152
50.1.3 Types of Rescue Robots	1153
50.2 Disaster Characteristics and Impact on Robots	1153
50.2.1 Categories and Phases of Disasters	1154
50.2.2 Natural Disasters	1155
50.2.3 Suitable Robot Technologies for Natural Disasters	1155
50.2.4 Manmade Disasters	1156
50.2.5 Suitable Robot Technologies for Manmade Disasters	1156
50.3 Robots Actually Used at Disasters	1156
50.3.1 2001 World Trade Center, United States	1157
50.3.2 2005 La Conchita Mudslide, United States	1158
50.3.3 2005 Hurricanes Katrina, Rita and Wilma, United States	1158
50.3.4 2006 Sago Mine Disaster, United States	1159
50.3.5 Post-Disaster Experimentation	1160
50.3.6 Search and Recovery	1161
50.4 Promising Robots and Concepts	1161
50.4.1 Alternative Ground Rescue Robot Designs	1162
50.4.2 Aerial Rescue Robots	1163
50.4.3 Unique Concepts of Operations	1164
50.5 Evaluation and Benchmarks	1165
50.5.1 Computer Simulations for Rescue Robotics	1165
50.5.2 Physical Test Beds	1166
50.5.3 Standards Activity	1167
50.6 Fundamental Problems and Open Issues	1167
50.6.1 Mobility	1168
50.6.2 Communications	1168
50.6.3 Control	1169
50.6.4 Sensors	1169
50.6.5 Power	1169
50.6.6 Human-Robot Interaction	1169
50.6.7 Evaluation	1171
50.7 Conclusions and Further Reading	1171
References	1171

Rescue robots serve as extensions of responders into a disaster, providing real-time video and other sensory data about the situation. They are an emerging technology, and have not yet been adopted by the international emergency response community. As of 2006, they have been used in only four disasters in the United

States (World Trade Center, and hurricanes Katrina, Rita, and Wilma), where they were still viewed as a novelty. However, rescue robots are seeing some use in local incidents. For example, several fire rescue departments in Japan and the United States routinely use small underwater robots for water-based search and

recovery, a ground robot has been used for a mine explosion in the United States, and interest in the use of aerial vehicles for wilderness search and rescue is growing. The general lack of adoption is to be expected since the technology is new, and the concept of operations of novel technologies as well as the refinement of the hardware and software coevolution will

take time. Rescue robot applications are often similar enough to military operations that the same platforms can be adapted; however, some rescue tasks are significantly different than their military counterpart, some tasks are unique to rescue, and the human–robot interaction for civilian response diverges from military patterns of use.

50.1 Overview

Disaster response is always a race against time, to move as fast as possible to reach all potential survivors and yet move slowly enough to avoid creating additional collapses, damage, or risk to rescuers and victims. The primary motivation is to save lives; robots can assist in meeting this goal either by interacting directly with victims or structures or automating support activities.

50.1.1 Motivation

Historically, the push for rescue robotics started in 1995 as outcomes of the tragic loss of life in the Hanshin–Awajii earthquake in Kobe, Japan, and the bombing of the Murrah federal building in Oklahoma City, United States [50.1]. The advances in robotics and artificial intelligence in the early 1990s supported the moral imperative for roboticists to reach out and help. As a result, research efforts began in individual laboratories. Two mobile robot competitions (the [AAAI](#) mobile robot competition in the United States, and the RoboCup rescue league internationally) were started shortly thereafter to engage the scientific community in rescue research.

The 2005 World Disasters report [50.2] suggests just how many lives have been, and will be, impacted by urban disasters. Over 900 000 people were reported killed from 1995 to 2004, with the total amount of disaster-related damage estimated at 738 billion US dollars. Of the victims in urban disasters, only a small fraction may actually survive. Consider from [50.3, 4] that the majority of survivors (80%) of urban disasters are surface victims, that is, the people lying on the surface of the rubble or readily visible. However, only 20% of survivors of urban disasters come from the interior of the rubble, yet the interior is often where the majority of victims are located, providing motivation for robots that can explore deep within collapses. The mortality rate increases and peaks after 48 h, meaning that survivors who are not extricated in the first 48 h after the event are unlikely to survive beyond a few weeks in the hospital.

50.1.2 Rescue Robot Tasks

While the overall motivation for rescue robotics is to save lives, the motivation for specific robot designs and capabilities depends on their potential tasks. The types of tasks that have been proposed for rescue robots are described below.

Search is a concentrated activity in the interior of a structure, in caves or tunnels, or wilderness and aims to find a victim or potential hazards. The motivation for the search task is speed and completeness without increasing risk to victims or rescuers.

Reconnaissance and mapping is broader than search. It provides responders with general situation awareness and creates a reference of the destroyed environment. The goal is speedy coverage of a large area of interest at the appropriate resolution.

Rubble removal can be expedited by robotic machinery or exoskeletons. The motivation is to move heavier rubble faster than could be done manually, but with a smaller footprint than that of a traditional construction crane.

Structural inspection may be either conducted on the interior (e.g., to help rescuers understand the nature of the rubble in order prevent secondary collapses that may further injure survivors) or on the exterior (e.g., to determine whether a structure is safe to enter). Robots provide a means of getting structural sensor payloads closer and in far more favorable viewing angles.

In situ medical assessment and intervention are needed to permit doctors and paramedics to interact verbally with victims, visually inspect the victim or apply diagnostic sensors, or to provide life support by transporting fluids and medication through narrow tubing during the four to ten hours that it usually takes to extricate a victim. The lack of medical intervention was a major problem at the Oklahoma City bombing [50.5].

Medically sensitive extrication and evacuation of casualties may be needed to help provide medical as-

sistance while victims are still in the disaster area, also known as the hot zone. In the case of a chemical, biological, or radiological event, the number of victims is expected to exceed the number that can be carried out by human rescuers in their highly restrictive protective gear; this makes robot carriers attractive. Since medical doctors may not be permitted inside the hot zone, which can extend for kilometers, robot carriers that support telemedicine may be of huge benefit.

Acting as a mobile beacon or repeater to extend wireless communication ranges, enable localization of personnel based on radio signal transmissions by providing more receivers, and to serve as landmarks to allow rescuers to localize themselves.

Serving as a surrogate for a team member, such as a safety officer or a logistics person. In this task, the robot works side-by-side with rescuers, for example, a group breaching rubble deep within the interior of a disaster may have difficulty using a radio to request additional resources because of noise. However, a team member outside of the rubble can see and hear through the robot the state of progress and anticipate needs. The objective is to use robots to speed up and reduce the demands of tasks, even if they are done by humans.

Adaptively shoring unstable rubble to expedite the extrication process. Rubble removal is often hindered by the need to adopt a conservative pace in order to prevent a secondary collapse that might further injure a trapped survivor.

Providing logistics support by automating the transportation of equipment and supplies from storage areas to teams or distribution points within the hot zone.

Some of the above tasks are similar to tasks for military robots, especially search and reconnaissance and mapping, but many are unique or have a different flavor. For example, structural inspection, rubble removal, and adaptively shoring rubble are rescue specific. Tasks such as casualty extraction appear to be similar but are significantly different. Consider that a wounded soldier is unlikely to have a spinal-cord injury and is likely to be in a space large enough for a human to work in, so a robot entering the area and dragging the soldier to safety is appropriate. However, a crushed victim of a building collapse is physically trapped or pinned in a small space, requiring rubble removal, and the victim's spinal

cord must be immobilized before extraction; clearly victim extraction in the search-and-rescue domain is more challenging.

50.1.3 Types of Rescue Robots

Rescue robots are needed to help quickly locate, assess, stabilize, and extricate victims who cannot be easily reached. They typically do this by extending the rescuers' ability to see and act. On the ground, small unmanned ground vehicles (UGVs) can enable rescuers to find and interact with trapped victims in voids that are too small or too dangerous for human or canine entry. Large UGVs can accomplish tasks such as removing large rubble faster than humans. In the air, unmanned aerial vehicles (UAVs) robots extend the senses of the responders by providing a bird's eye view of the situation. In the water, unmanned underwater vehicles (UUVs) and surface vehicles (USVs) robots can similarly extend and enhance the rescuers' senses.

Rescue robots can be broadly categorized into types based on *modality* and *size* [50.6], though other taxonomies that mix modality, size, and task have been proposed [50.7]. There are four modalities of robots: *ground*, *aerial*, *underwater*, and *water surface*. The modality impacts on the basic design and capabilities of the robot. Within each modality, rescue robots can be further described as one of three sizes: *man-packable*, *man-portable*, and *maxi*. The size of the robot impacts both on the tasks for which it is suited and how soon after a disaster it might be used. In order to be man-packable, the entire robot system, including the control unit, batteries, and tools, must fit into one or two backpacks. Man-packable robots are more likely to be used immediately after a disaster since they can be carried by responders over debris and up and down ladders into the core of the disaster, while larger equipment must wait for paths to be carved out. The next larger size is man-portable; these are robots that can be carried a short distance by two people or on a small all-terrain vehicle. Man-portable robots can be used as accessibility within the hot zone improves or outside the hot zone for logistics support. Maxi-sized robots require trailers or other special transportation logistics. This inhibits their insertion into the hot zone, unless they are being used on the exterior of the rubble.

50.2 Disaster Characteristics and Impact on Robots

The type of disaster influences the choice of robot platforms and payloads. *Natural disasters* generally span

large geographic areas, making a bird's eye view from a UAV invaluable in establishing situation awareness and

determining areas or individuals which need immediate evacuation or assistance [50.8]. *Manmade disasters* are geographically concentrated with the most vital aspects of the disaster invisible beneath the rubble. Small ground robots that can enter deep into the interior of the rubble, and large robots, which can help remove rubble, appear to hold the most promise for manmade disasters. Note the focus on the search for victims and on general information gathering. This focus implies that communications plays a major role in the design of search-and-rescue robot systems and the impact of the environment on wireless communications is an important consideration.

Independently of the type of disaster, search and rescue is very demanding both in terms of robotic capabilities and working conditions for the operator [50.9]. The robot is, by definition, operating in a harsh, mobility-challenging environment. The presence of abrasive dust and water, the corrosive effects of wet cement, and the wide range of obstacles in the environment serve to accelerate the wear and tear on a robot [50.6, 10]; therefore, the adage *simple is better* is especially true in rescue robotics. The disaster also places demands on the robot operator that will likely impact performance. Certainly, the operator will be stressed by the time criticality and life-saving urgency of the mission. Other sources of stress include: the operator will not be able to keep the robot in line of sight (which is a more favorable operating regime), perception to the robot is computer-mediated and therefore cognitively fatiguing, and the operator is likely to be sleep-deprived and may be at some personal risk.

50.2.1 Categories and Phases of Disasters

An incident may be *local* or it may be a true *disaster*; a disaster exceeds local resources or expertise and requires specially trained teams from outside the immediate agency to be involved. Disaster operations consist of four phases: *preparedness*, *prevention*, *rescue*, and *recovery*. Preparedness and prevention are *pre-incident activities*, whereas rescue and recovery are *post-incident tasks*. Rescue is also distinct from the recovery phase of a disaster operation; recovery seeks to mitigate longer-term threats to life and damage to property and to extract the dead.

Rescue is the broad term applied to activities immediately following a critical incident that directly deal with immediate threats to the survivability of those impacted by the event. This includes locating, assessing the medical condition, stabilization, and extrication of

survivors. *Rescue* is often used interchangeably with the term *search and rescue*, which actually refers to the teams and activities that work in the field, whereas rescue is frequently used by the public to connote the larger disaster management activities needed to support the teams and victims. *Search* refers to activities related to finding survivors; while *rescue* refers to the activities related to extricating survivors. In general, the typical process of search and rescue takes two tracks: *strategic* and *tactical*. Strategic operations focus on mission planning and coordination for the entire enterprise, which may involve robots as mobile sensor, communication networks, or logistics support.

Within search and rescue, there are numerous specialties and these specialties may impact on the design or use of a robot. Of these, *urban search and rescue* (abbreviated as **US&R** or **USAR**) has been the subject of the majority of work in rescue robotics. Urban search and rescue takes its name from its focus on the aftermath of urban structures collapsing around people. These structures may collapse for natural reasons such as earthquakes, hurricanes, and flooding or they may stem from manmade causes such as structural failures or terrorism. However, there are other types of search and rescue, though these are generally associated with local incidents. *Wilderness search and rescue* tracks people in the outdoors, such as lost hikers or people buried by avalanches. *Water-based search and rescue* deals with situations such as saving victims of floods or high currents (also known as *swift water rescue*) or in the aftermath of traffic accidents where victims are trapped in cars that have plunged into a river or bay.

In order to understand how to apply robotics to disaster response, it is helpful to understand the general pattern of activity, which can be summarized as [50.11, 12]:

1. Responders become aware of the existence of victims. This awareness may be generated by information from family, neighbors, and colleagues, an understanding of demographic patterns (e.g., at night, apartment buildings will be heavily occupied, while during a work day, office buildings will be occupied), or by a systematic search.
2. The response command staff attempt to understand the disaster site. They investigate the site for conditions such as hazardous materials, the risk posed to the rescuers themselves, any pending threats to trapped victims, and resource restrictions such as barriers to transporting resources to the site, nearby

usable equipment and materials that can be exploited, and any other barriers to a timely rescue.

3. The command staff plans the operations.
4. Search and reconnaissance teams are sent to map the situation and assess environmental conditions. Accurate estimates of the need for emergency medical intervention are highly desirable in order to optimize allocation of medication personnel in the field and to prepare ambulances and hospitals. (In the case of the Kobe earthquake, this stage took the longest time [50.11].)
5. Excavation of rubble to extract victims begins. Note that removal of rubble in search and rescue differs from construction removal of rubble, because the safety of the victims is the top priority.
6. Responders gain access to victims and apply emergency medicine in situ.
7. Victims are transferred to hospitals.
8. Field teams report activities periodically, usually at the end of the shift, and the command staff modifies or replan accordingly.

Robots are particularly needed for *tactical search and rescue*, which covers how the field teams actually find, support, and extract survivors. Tactical search and rescue typically begins independently of strategic planning [50.13]. As discussed in [50.12], the first responders on the scene, policemen and firemen, as well as civilians immediately assist with extracting survivors. Regional search-and-rescue teams are deployed and arrive within a few hours, while strategic operations are only beginning to be set up. Based on the condition of the structure and the size of voids, rescuers may enter the rubble, typically using a right-wall-following algorithm. In order to locate inaccessible victims not visible on the surface of rubble, rescuers working in teams of two typically bring in canine teams to smell for the presence of survivors, or use pole-mounted search cameras. Search cameras and boroscopes can usually extend visibility into the rubble by several meters, depending on the irregularity and turns in the voids. If a team has any indication of a survivor, another team is called over to verify the finding. If the finding is verified, and the victim is presumed to be alive, extrication begins. Rescue teams work systematically through a structure and mark each void as to when it was searched and what the findings were.

50.2.2 Natural Disasters

Natural disasters, such as earthquakes, tsunamis, hurricanes and typhoons, volcanoes, avalanches, landslides,

and floods, present many challenges for rescue robots. Natural disasters are usually geographically distributed, perhaps affecting a 200 km or more radius around the epicenter of the event. The sheer size of the affected area presents many challenges to the emergency response. The primary impact of natural disasters is on residences, light commercial buildings, sea walls and canals, and transportation and communications infrastructure. This means rescuers have thousands of structures to check quickly for survivors, but those structures will be fairly small and amenable to manual and canine search. Besides the sheer volume of structures to check, communication disruptions prevent rescuers from getting timely information as to the state of transportation access and the general needs of an area. However, designing robots to meet these challenges is important because natural disasters provide the most hope of a large number of survivors. Uninjured survivors may simply be stranded and can survive for up to 72 h.

As described in [50.8], responders are often left with no choice but to break up into small teams and begin searching in the largest centers of population with rescue on a *first-come first-served* basis. Because of the lack of pervasive communications, these teams and squads must work independently and typically with no real-time access to information being gathered by governmental agencies. Survivors may have hidden themselves in closets or in the attic, and therefore be difficult to detect. The risks to rescuers include live or suddenly re-energized electric lines, gas leaks, contamination from sewage, and victims who may try to protect themselves from perceived looters.

50.2.3 Suitable Robot Technologies for Natural Disasters

Natural disasters present unique opportunities and problems for robots. The major missions are to provide situational awareness of the distribution and degree of need of survivors (e.g., who needs immediate evacuation, who can remain in place longer), identification of routes of entry for responders, and assessment of conditions that may present a further immediate threat to life [50.8, 14]. These missions generally require coverage of large areas with information ideally provided immediately to the team of responders. As such, these types of missions favor the use of UAVs, particularly man-portable and man-packable models that can be launched as needed from small clearings and provide real-time video directly to the responders. Rotary-wing UAVs are useful even in densely forested rural areas to

enable the assessment of rural areas and road damage. Both types of vehicles may be flown at night, but this would be subject to local airspace regulations.

It is important to remember that disasters often have a water-related component. Most of the world's population lives near a bay or river and relies on bridges and sea walls. **USVs** are expected to be of more benefit than **UUVs** in inspecting this infrastructure for several reasons [50.15]. They can be launched from a distance, and because they provide a view above and below the water line of bridges, sea walls, docks, etc. **UUVs** are difficult to control in swift water and may become easily trapped by debris in the aftermath of flooding or a cyclonic event.

As a whole, ground robots may not be as immediately useful as **UAVs** and **USVs** for natural disasters [50.8, 16]. Ground robots are handicapped by the number of dwellings and light commercial buildings that must be explored at least as rapidly as a human team, compounded by the need to often break down a door or window in order to insert the robot. It is unlikely that ground robots will be able to compete with canines and their ability to smell the presence of survivors without having to break and enter intact dwellings.

50.2.4 Manmade Disasters

In comparison to natural disasters, manmade disasters (such as a terrorist bombing or serious accident) occur in a small area. The challenge is often not how to see the entire external extent of the damage, but rather to see what is not visible: the interior of the rubble, the location and condition of survivors, and the state of potentially dangerous utilities (e.g., electricity, gas lines) [50.6, 17]. The communications and power infrastructure usually exist within a 10 km range and cell

phones generally work outside of the collapsed area. Voids in the rubble may be irregular in shape and vertical in orientation. Wireless communications in the interior of the rubble is unpredictable, and generally nonexistent due to the large amount of steel within commercial structures, but the combination of irregular voids and sharp rubble do not favor the use of fiber optic cables. Visibility is difficult as there is no lighting and everything may be covered with layers of gray dust, further hampering recognition of victims, potential hazards, or accurate mapping. The interiors may be wet or contain standing water due to water lines, sewers, and sprinkler systems. Survivors are more likely to be in dire need of medical attention.

50.2.5 Suitable Robot Technologies for Manmade Disasters

The attractiveness of robots for manmade disasters stems from their potential to extend the senses of the responders into the interior of the rubble or through hazardous materials. **UGVs** are expected to be essential for building collapses, attacks such as the 1995 sarin release in the Tokyo subway, and responses to radiological disasters such as Chernobyl and Three Mile Island, while **UAVs** are key for mapping external plumes and spills of toxic chemicals. **UGVs** are expected to be the dominant modality for victim rescue in a building collapse, though other modalities might help with long-term recovery. This strongly suggests that **UGVs** should be waterproof or at least highly water resistant, since the interior of the rubble is likely to have some water present from the sprinkler and sewage systems. The small size and irregular shape of voids in the rubble also suggests that **UAVs**, no matter how small, will be unlikely to have enough space to fly in and navigate the interior.

50.3 Robots Actually Used at Disasters

While many types of robots have been proposed for search and rescue, only a few have actually participated in a rescue or been allowed to operate on site after a disaster for testing purposes; all of these have been teleoperated. Six of the seven disasters that have employed robots have occurred in the United States, with one in Japan (Table 50.1). The majority of deployments have fortunately been by scientists at the Center for Robot-Assisted Search and Rescue (United States) or the International Rescue System Institute (Japan), providing

crucial information about robot design, concepts of operation, and human–robot interaction. In terms of ground robots, small tracked platforms dominate but there have been promising results with a snake robot fielded at the Niigati Chuetsu earthquake, as described below. Man-portable and man-packable fixed- and rotary-wing aerial platforms also have been useful at disasters. Many state National Guards in the United States own the Predator **UAV**. Unmanned surface vehicles have been used once with good results. Small **UUVs** such as the VideoRay

Table 50.1 Tables showing the modalities and sizes of robots deployed to actual disasters or used for experimentation at the disaster site

Disaster Deployment		UAV	UGV	USV
2001	World Trade Center, New York, USF		Man-packable Man-portable	
2005	La Conchita, California mudslides, USF		Man-packable	
2005	Hurricane Katrina, USA	Man-packable fixed wing Man-packable rotary wing	Man-packable	
2005	Hurricane Rita, USA	Maxi fixed-wing		
2005	Hurricane Wilma, USA	Maxi fixed-wing		
2006	Sago Mine, West Virginia, USA		Maxi	
Post-Disaster Experimentation		UAV	UGV	USV
2001	World Trade Center, New York, USF		Man-packable Man-portable	
2004	Niigati Chuetsu Earthquake	Man-portable snake		
2005	Hurricane Wilma, USA	Man-portable fixed-wing		Maxi
2005	Hurricane Katrina, USA	Man-packable rotary-wing		

are commonly used internationally to help find bodies after a car accident or drowning. Only three agencies are known to own ground rescue robots, the Tokyo Fire Department in Japan, the New Jersey Task Force 1 State Urban Search and Rescue team in the United States, and the United States Mine Safety and Health Administration. There is no known disaster where a robot actually found a survivor, though robots have generally received high marks for completing their search missions and finding remains.

50.3.1 2001 World Trade Center, United States

Unmanned ground vehicles were used at the World Trade Center 9/11 disaster in New York City, United States [50.10]. The disaster occurred on 11 September 2001 with terrorists crashing planes into the twin towers, which subsequently collapsed. Another building in the complex was sheared in half, and a fourth building burned down. Close to 3000 people and firefighters were killed immediately. A massive search-and-rescue response was conducted in the hope of finding trapped survivors in stairwells and basements covered under the pancaked debris of 110 storeys of collapsed steel structure.

Three species of small UGVs (Inuktun micro-VGTV, Inuktun micro-Tracks, and Foster-Miller Solem) were used initially to actively explore the rubble of the twin towers, Tower One and Tower Two, plus the partially demolished Tower Four. Later the Foster-Miller Talon was also used to inspect the building founda-

tions. The UGVs were all affiliated with the DARPA Tactical Mobile Robots program, and were fielded under the direction of the Center for Robot-Assisted Search and Rescue (CRASAR) in conjunction with the state of New York State Emergency Management department and the New York Department of Design and Construction. Seventeen species of robots were available, but only the most portable, ruggedized, and user-friendly were actually selected for use by the responders.

The first robots arrived on scene by 16:00 on 11 September 2001 and tethered robots were put into service around midnight. Robots were used through to 2 October 2001, when the last robot broke. The robots and operators were deployed with several Federal Emergency Management Agency teams, especially Indiana Task Force One, through until 21 September 2001. The robots were used to explore voids with less than 1 m diameter that a person or dog could not enter or voids less than 2 m diameter that were still on fire. The robots found approximately 10 sets of remains in rubble 7–20 m below the surface but no survivors. Only one robot was lost, a Solem, which lost wireless communications, stopped in place, and the safety rope broke during retrieval; note that the density of the rubble significantly interfered with wireless networks. Later in September, as rescue became improbable, larger man-portable Foster-Miller Talon robots began to be used to explore the stability of the basement, especially the slurry wall. The slurry wall was a foundation shared by many other buildings in that part of New York and damage could create further structural difficulties. The

New York (City) Department of Design of Construction actually breached holes in likely locations of the perimeter of the basement in order to insert the larger robots.

The lessons learned fall into two categories: *design of robots* and *human–robot interaction*. These are summarized below, following [50.10]. Robot platforms will be operating in narrow, vertical spaces that pose communications interference. Each robot will need a safety rope and, if wireless, the wireless system should be replaced with a communications (fiber-optic) tether sturdy enough to act as a safety rope. Robots should be invertible, since the confined spaces often do not permit sufficient room for self-righting, and with sufficient waterproofing to permit decontamination and to operate in water, rain, and snow. The minimum payload is a color camera and two-way audio, while an operator control unit should contain record and playback video capabilities. Thermal imaging was not of use at the World Trade Center due to the overall heat in the voids, but is generally desirable.

50.3.2 2005 La Conchita Mudslide, United States

Ground robots were used unsuccessfully at the La Conchita mudslide [50.18]. On 10 January 2005, at the small town of La Conchita outside of Los Angeles, a sudden mudslide destroyed 18 houses, killed ten people and left searchers looking for six people missing for several days (the missing persons were out of the state on vacation). While this was not a mass disaster, it did extend beyond a local incident and engaged regional and national response teams.

CRASAR responded with an American Standard Robotics VGTV Extreme, a waterproof upgrade of the polymorphic Inuktun micro-VGTV used extensively at the World Trade Center. The robots were used twice and failed within two minutes and four minutes, respectively. The obvious reason for the failure in both cases was poor mobility. The robot was initially directed to search a crushed house where a canine team had indicated the possibility of a victim. The robot entered the house via a narrow gap between the foundation and flooring, but immediately failed when a root in the wet soil became lodged in the left track and it came off. In the second run, the robot was inserted into the intact second story of a house damaged on the lower floor to test a vertical entry. The robot detacked once again, this time due to the compliance of the thick shag

carpeting. Though a poor track design (originally intended for movement on smooth ventilation ductwork) was the ultimate cause of failures, the severity of the design flaw was undetected by both the manufacturer and CRASAR due to the lack of adequate testing under realistic conditions.

The lessons learned for robotics from this disaster were that ground mobility remains a major challenge and that adequate testing is important. Open track designs appear to be unsuitable for rescue; they are too susceptible to detacking and vulnerable to debris becoming wedged in the tracks. Testing in diverse soil and interior terrain conditions is important, adding emphasis to the need for adequate standards.

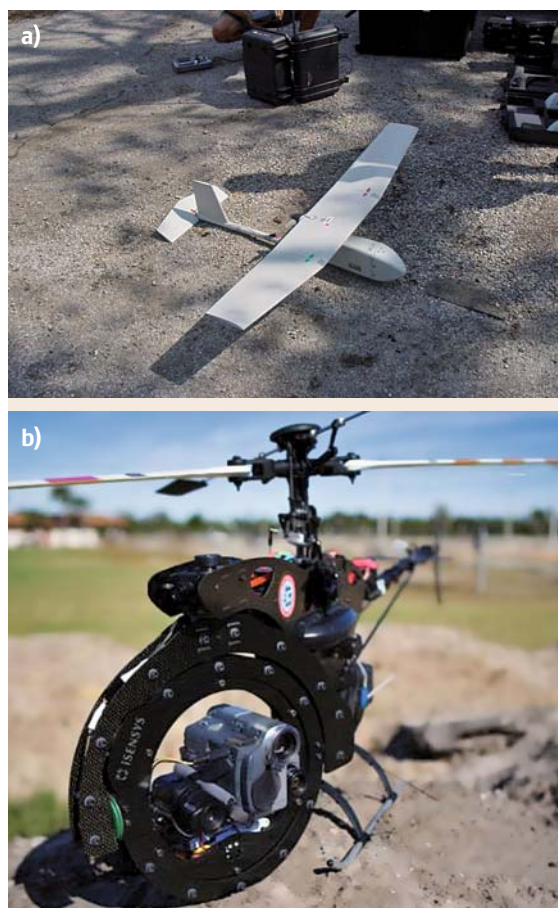


Fig. 50.1a,b Man-packable UAVs used to search portions of Mississippi during the hurricane Katrina response: (a) an anonymous fixed-wing UAV and (b) an iSENSYS IP3 rotary-wing UAV (courtesy of CRASAR)

50.3.3 2005 Hurricanes Katrina, Rita and Wilma, United States

State and military unmanned aerial vehicles were used in the aftermath of several hurricanes in the United States, along with a ground robot. In 2005, the southern Gulf Coast of the United States was battered by a series of hurricanes from June through November. Hurricane Katrina was the worst hurricane in the history of the United States. It made its primary landfall near New Orleans on 29 August 2005, devastating the states of Louisiana and Mississippi and portions of Alabama, impacting an area of 200 km², costing nearly 2000 lives, and leaving behind 80 billion US dollars in damage. Hurricanes Rita and Wilma were less extensive and damaged parts of Texas, Louisiana, and Florida.

The 2005 hurricane season saw the first use of UAVs for a large-scale disaster [50.16]. During the hurricane Katrina response, two UAVs were used by CRASAR in Mississippi as part of the Florida State Emergency Response Team to search rural regions cut off by flooding and downed trees. The UAVs are shown in Fig. 50.1: a battery-powered fixed-wing and a battery-powered rotary-wing (a Like90 T-Rex miniature helicopter modified for stability in high winds) UAVs. An internal-combustion Silver Fox was used later in the week over New Orleans by the Department of Defense to help identify which areas were still in need of assistance. All systems were under 2 m in their characteristic dimension, easily dismantled and transportable, and flew below regulated airspace, with procedures spontaneously developed to ensure safety and avoid collisions with manned aircraft. The UAVs were tasked for surveying, providing information directly to the responders rather than waiting for the data from unmanned helicopter flights to be processed and then manually transported to the responders in the field. In hurricanes Rita and Wilma, the Texas and Florida state National Guards flew much larger Predator UAVs in regulated airspace to provide timely information to strategic decision makers. The Predator class of UAVs are physically larger, require more people to operate, and a large take-off and landing zone, and must be coordinated with airspace operators since they operate at the same altitudes as manned aircraft.

An American Standard Robotics VGTV Extreme robot, the same model that failed at La Conchita, was used by Florida Task Force 3 State Urban Search and Rescue Team to successfully investigate the first floor of an apartment building in Biloxi, Mississippi, which was unsafe for human entry. The team did not have a canine

and was unsure if a trapped victim would be able to shout in reply to an audio challenge, so the decision was made to deploy the robot. Unlike La Conchita, the area of operation was favorable and the interior was smooth linoleum. The robot showed that no one was trapped inside.

The lessons learned during the 2005 USA hurricane season were primarily related to how large, geographically distributed disasters influence the choice of robot modality and how they can be used. The disasters highlighted the need for miniature aerial vehicles which could provide tactical teams with viewpoints of 1–10 km on demand or viewpoints that had not been possible before (e.g., from a miniature helicopter) [50.19]. Autonomous waypoint navigation of fixed-wing platforms was available but was not used; instead responders were much more interested in actively directing the UAVs. This suggests that the ultimate control regime will not be full autonomy but rather enabling a responder to use the robot as if it were an extension of their person.

50.3.4 2006 Sago Mine Disaster, United States

A maxi-class ground robot was used at the Sago Mine disaster in the United States. On 2 January 2006, a mine explosion in Sago, West Virginia killed 12 miners and severely injured another. It took two days to reach the victims, who were approximately 3 km from the surface; the primary obstacle was not physical debris but the need to operate in an environment saturated with carbon monoxide and methane. The Sago Mine disaster is considered a disaster, despite the relatively small number of people trapped in a confined area, because rescue teams and equipment had to be deployed from other states and from the Mine Safety and Health Administration (MSHA).

A large Remotec ANDROS Wolverine robot was deployed by the Mine Safety and Health Administration. The Wolverine class of robot is primarily designed for bomb-squad applications, where its slow speed and heavy weight (over 500 kg) is not a disadvantage. The robot had been previously modified with an enclosure rendering it intrinsically inert in the presence of explosive gases and a fiber-optic communications tether added to improve communications. Even with its slow speed and limited agility, the robot has the advantage of being able to enter a mine faster than a person in a contained suit and also being less likely to create an explosion. Unfortunately, the robot, nicknamed V2, was only able to



Fig. 50.2 The V2 robot used at the 2006 Sago Mine Disaster (courtesy of the US Mine Health and Safety Administration)

penetrate about 700 m into the mine before getting stuck on rails.

The lessons learned from the use of a UGV at the Sago Mine disaster highlighted again the need for better mobility, communications, and navigational autonomy. It illustrated the need for robots to enter areas that are unsafe for humans, but also that simply *wrapping* a robot with an explosion proof enclosure is a stopgap measure; a new class of robots for operating in explosive environments is necessary. Communications remains difficult, and the fiber-optic tether is a vulnerable component. While the fiber-optic tether was not severed during the disaster deployment, it had been damaged in prior tests and reopenings of mines. The ability to maintain reliable wireless communications with more agile robots remains a top priority in underground rescue. Navigation via teleoperation is a known challenge and often distracts the operator from larger search and assessment issues.

50.3.5 Post-Disaster Experimentation

In many cases, the real hope of rescue has passed before robotics equipment can be requested and deployed. Fortunately, forward-thinking response personnel may allow roboticists to work in the disaster zone (hot zone) during the later stages of the rescue or the recovery. In these deployments, the robots were not actually used for rescue operations but rather for experimentation, allowing the response community to evaluate and comment on rescue robot progress as well as rescuers to collect valuable data. Operations in post-disaster conditions are much more realistic than the best test beds.

At the 2001 World Trade Center disaster, CRASAR teleoperated an iRobot Packbot and a SPAWAR Urobot in a collaterally damaged building and parking garage near the main disaster site [50.10]. While the robots worked largely as expected, one surprise was the negative impact of water and dust on stair climbing and general mobility. The water from the sprinkler systems and the dust made surfaces very slippery.

In 2004, the International Rescue System Institute (IRS) inserted a serpentine robot into a house damaged during the Niigata Chuetsu earthquake (Fig. 50.3). This was the first snake robot used at a disaster site and showed the promise of biomimetic alternatives to tracked platforms. The first quake occurred on 23 October 2004, and was the largest to have hit Japan since the Hanshin Awaji earthquake in 1995. Thirty-nine people were reported to have died and massive damage was done to the transportation infrastructure. The robot, a variant of the Soryu III robot developed in collaboration between the Tokyo Institute of Technology and IRS, is 1.2 m long and weighs 10 kg, with a turning radius of 0.41 m, and a maximum speed of 0.37 m/s. The IRS Soryu carried a charge-coupled device (CCD) camera, an infrared camera (FLIR), and two-way audio as well as proprioceptive sensors. The robot was controlled through a tether that also served as a safety rope. In addition, the Soryu can support a CO₂ sensor that can detect human breathing, navigation sensors such as a laser range finder, and localization sensors, though it does not appear that this was tested. The testing in the debris concentrated on mobility.

Three days after hurricane Wilma in 2005, CRASAR deployed both a modified Like90 T-Rex and a man-portable, experimental unmanned surface vehicle (USV) to inspect seawall and bridge damage at Marco Island,

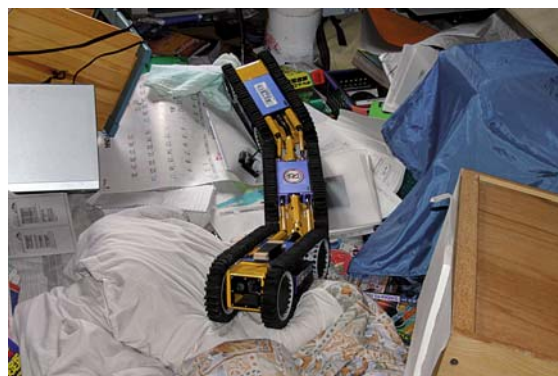


Fig. 50.3 IRS Soryu robot searching a house destroyed by the Niigata Chuetsu earthquake (courtesy IRS)



Fig. 50.4 Unmanned surface vehicle exploring the bridge to Marco Island, Florida (courtesy of CRASAR)

Florida [50.15]. The USV, shown in Fig. 50.4, carried a DIDSON acoustic camera, capable of penetrating through turbid waters and mud, as well as a camera above the surface. The UAV was used independently to investigate damage topside, but was very helpful to the USV crew in establishing where the USV was in relationship to piers and pilings; this strongly indicates the need for UAV-USV teams. The use of the USV near and underneath large structures hampered wireless communications and Global Positioning System (GPS) signals, posing additional demands.

Three months after hurricane Katrina in 2005, CRASAR used a small rotary-wing UAV, an iSENYS IP3, to document the structural damage to multistory commercial buildings along the Gulf Coast impacted by hurricane Katrina [50.16, 19]. The eight days of intense flying, which duplicated emergency response missions, showed that three people are needed to operate a UAV safely close to urban structures: a pilot, a mission specialist to run the payload and take pictures, and a flight director to serve as safety officer and maintain overall situation awareness. The post-hurricane-Katrina effort confirmed the concept of operations from hurricane Wilma, where a mission consists of a short flight (five to eight minutes) on each face



Fig. 50.5 A small UUV used for underwater inspection (courtesy of CRASAR)

of the building, allowing the pilot and flight director to keep the vehicle in line of sight. This concept of operations fosters safety and reduces cognitive fatigue for the pilot. Although it is likely that teleoperated line-of-sight in the United States will be mandated, the close proximity to the building (1–3 m) indicates the need for semiautonomy, especially guarded notion, to assist the pilot and prevent collisions.

50.3.6 Search and Recovery

While ground and aerial rescue robots are new, very small underwater robot cameras such as that shown in Fig. 50.5 have been routinely used by law enforcement and fire rescue teams for over a decade to find submerged cars that have gone into the water. These operations are generally considered search and recovery, rather than rescue, because the victims have drowned and are beyond revival by the time the specialized equipment can arrive. The UUVs are miniature torpedo-like devices with cameras teleoperated by a person on the surface. The problems facing UUVs are sensing through the turbidity of the water and control of devices in strong currents that prevent them from being used in swift water.

50.4 Promising Robots and Concepts

Search and rescue is a demanding application and it is reasonable to expect that new types of robots will evolve to meet the challenge. Numerous speculative designs for unmanned ground and aerial vehicles

have been proposed, and representatives that have been tested are described below. While approaches to search and rescue have mainly been presented in terms of single-robot activities, teams of robots proffer inter-

esting future scenarios where teams of robots swarm and cooperate to accomplish their mission. Robots need not look like robots in order to be useful for search and rescue; unattended ground sensors, blimps, kites, and smart shores using the principles of robotics and artificial intelligence are poised to make a difference in the near future. In addition, basic research in legged robots (Chap. 16), wheeled robots (Chap. 17), micro/nanorobots (Chap. 18), multiple-robot systems (Chap. 41), networked robots (Chap. 42), general underwater robots (Chap. 44), and aerial vehicles (Chap. 45) are expected to yield more concepts for rescue robotics.

50.4.1 Alternative Ground Rescue Robot Designs

Polymorphic tracked vehicles represent the state of the practice of ground rescue robots. These tracked vehicles have been primarily used for navigation and sensing missions, ignoring missions that require manipulation. Wheeled platforms are severely limited by the roughness of the terrain and the need to overcome obstacles, steps, ramps, but combination wheeled and tracked vehicles are commercially available. Small tracked robots with manipulators have been developed for bomb-squad activities and appear well suited for transfer to rescue applications. In addition, many alternatives to tracks have been proposed. Of these, serpentine (or *snake*) robots and legged vehicles appear the most promising. Jumping or rolling robots have been proposed to cope with rough terrains, but their effectiveness still needs to be demonstrated.

Robots with manipulators (such as that seen in Fig. 50.6) extend the capabilities of ground vehicles by allowing the robot to sample the environment, interact with survivors, move light obstructions (manipulators typically cannot lift heavy objects), and add unique camera viewpoints. Operators of the Foster–Miller Talon robot at the World Trade Center disaster used the camera mounted on the robot arm to peer over railings and inspect the basement substructure below. Manipulator arms are often used as camera masts, allowing the operator to see more of the environment, examine the surfaces of desks, tables and counters, and to see if a robot is stuck or disabled. However, adding a manipulator comes at a cost. The manipulator extends the volume of the robot, impacting on navigation; the arm is often at risk of being damaged in confined spaces by being hit on overhanging rubble. Manipulators also add to the control and mechanical complexity of the robot.



Fig. 50.6a,b Examples of tracked robots with manipulator arms that have been used for disasters or rescue competitions. (a) View from Foster–Miller Talon at the WTC of its arm (courtesy of CRASAR) and (b) telemAX by Telerob at the Rescue Robotics Camp (courtesy of R. Sheh)

Free serpentine robots such as the Soryu III used at the rubble at the Niigata–Chuetsu earthquake provide a fundamentally different style of mobility, while *fixed-based* snakes can be used in tandem with a more traditional platform as a highly flexible sensor manipulator [50.20]. Examples of both types of snake robots are shown in Fig. 50.3 and Fig. 50.7. Serpentine robots are more correctly known as hyperredundant mechanisms [50.21]. Free snakes propel themselves either by direct propulsion, making direct contact with the ground, or undulation, wriggling the internal degrees of freedom of the robot. Snake robots present some of the most demanding challenges for roboticists [50.22]. In order to be truly deployable, much work is needed in design and actuation, as well as control. Autonomous gaits need to



Fig. 50.7 A CMU fixed-base snake being tested at a facility in California (courtesy H. Choset)

be developed in conjunction with a sensor skin that will let a snake move in confined spaces out of the line of sight of an operator.

In order to overcome the difficulties posed by unknown terrain, legged robots and crawler robots have been proposed. In addition, some types of crawler robots can climb walls and reach locations that would otherwise be very hard to reach. Legs are interesting because they exploit biomimetic principles. The RHex hexapod robot [50.23], proposed for search and rescue among other potential applications, is shown in Fig. 50.8 climbing random step fields. The legs on RHex are curved and rotate, yet duplicate the spring-like capabilities of insect legs. They illustrate how biologically inspired forms of locomotion do not have to look like animal legs to achieve the same effect. RHex has favorable mobility properties but the slapping motion of the legs may stir up large amounts of dust in interior building collapses and interfere with sensing. Crawler robots such as the Terminatorbot [50.24] use their arms or legs to pull themselves through the rubble. The Terminatorbot is designed to withdraw itself into a cylinder that can be inserted through one of the small boreholes commonly drilled by responders to get through walls, then open up and begin moving. Other types of crawlers include lizard- or gecko-like robots that adhere to walls; these types are promising but have not been tested for the dusty, wet, and irregular conditions found in disasters.

50.4.2 Aerial Rescue Robots

Aerial rescue robots represent the most advanced robotics technology in use, and new concepts continue to emerge. Aerial vehicles can be further subdivided into *fixed wing* (a.k.a. plane-like), *rotary wing* (helicopters),

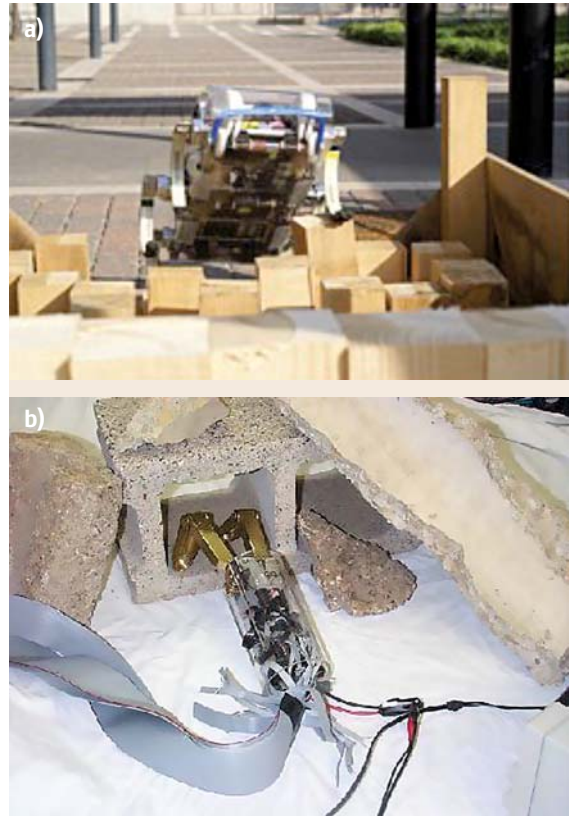


Fig. 50.8a,b Examples of legged and crawler robots. (a) Hexapod (legged) robot from the RHex project traversing a portion of the NIST test bed (courtesy of R. Sheh) and (b) Terminatorbot being tested in rescue test bed (courtesy of R. Voyles)

lighter-than-air (blimp), and *tethered* (kite) platforms. Fixed-wing UAVs typically travel long distances and can circle points of interest, while rotary-wing platforms can hover and require a small area to launch and land. Lighter-than-air vehicles may be tethered, similar to a kite. Underwater vehicles may also be subdivided into tethered and untethered platforms.

Large unmanned helicopters, such as the Yamaha R-Max, continue to be adapted for commercial rescue-and-recovery missions, including carrying heavy payloads capable of estimating the amount of rubble and debris generated by a disaster. Small fixed-wing platforms for tactical military use may have great benefit to the response community. Novel ideas include a plane the size of a person's hand that can fly indoors and planes with foldable wings, making it easier for responders to carry them. Novel rotary-wing designs are

also being proposed. Quadrotor helicopters appear far more stable and easier to pilot; a design that balanced the larger size of a quadrotor with an appropriate payload could make UAVs more assessable to non-pilots. Another exciting direction is hybrid platforms that can change from fixed-wing operations, covering large distances, to rotary-wing operations, flying near or inside buildings [50.25]. However, UAVs are drawing the attention of agencies that control airspace and may become heavily regulated in the future.

50.4.3 Unique Concepts of Operations

Current rescue robot scenarios have typically complemented or enhanced human capabilities, but within a fairly conventional, single-robot framework. New concepts of operations for exploiting rescue robots are being continuously identified. At this time, the major themes are *multi-robot teams* or *swarms*, robots forming or extending *communications and sensor networks*, *smart tools*, and *roboticized animals*. Past experience has shown that current ground robots are not appropriate for multi-robot team operations; relatively few voids occur that current robots could explore and these may be in different sectors assigned to different teams, so the idea of an operator managing 10 or more robots each actively investigating the rubble is speculative for now.

Teams of robots that cooperate with each other or work on the same objective have already been demonstrated for rescue activities. Homogeneous teams have been entered at the RoboCup rescue competition [50.26]. After hurricane Wilma, a UAV and USV team accomplished together what a single USV could not [50.15]. The small Inuktun robots used at the World Trade Center were actually intended to be the daughters (or joeys) of a marsupial (kangaroo) team, where a larger ground robot would deposit the smaller robots into voids [50.27]. (In reality, responders carried and inserted the robots.) Aerial vehicles that drop intelligent sensors and motes is another type of heterogeneous team. One of the most intriguing types of teams is swarms. Researchers such as [50.28] have discussed the possibility of using cost-effective, insect-sized robots to penetrate deep within a pancake building collapse and then signal the presence of a victim. Such teams can make use of collective intelligent behaviors from simple animal models that are far more complex than the behavior exhibited by the individuals. Consequently, robot teams can be composed of a large number of simple devices, thus yielding greater robustness to failure and distributed information gathering. One of the key features of swarm

approaches is that they can scale up easily. The insect swarm scenario leaves hard problems like control, sensing payloads, localization of the victim, and communications to the imagination, but is certainly a worthy concept. However, some of the search algorithms used by insects may be adapted to single robots, for example win-shift win-stay sampling exhibited by bees may be useful for search [50.29].

Robots, stationary and mobile, can facilitate the establishment of communications and sensor networks. As repeaters for mobile ad hoc networks, robots on the land, sea, and air can extend the range and throughput of wireless networks [50.30]. Aerial vehicles are particularly attractive as they can provide larger relay ranges while providing a bird's eye view of the disaster. However, a UAV does not always have to move: a tethered blimp or kite can support a sophisticated payload with no maintenance or support for days [50.31, 32].

Another concept is that of *smart* tools, particularly lifts that can help stabilize collapsed structures during extrication [50.33, 34]. Extrication is one of the most time-consuming activities in rescue. Rescuers must proceed cautiously when removing rubble in order to prevent secondary collapses or slides that would further injure the survivors. Experiments suggest that roboticized lifts or shoring mechanisms would be able to sense and respond fast enough to small movements in the rubble to adaptively maintain stability. [50.34]

The trend towards miniaturization of sensors and wireless communications has led some researchers to propose having search dogs carry roboticized cameras (dog-cams) or wiring rats with controllers. Attaching cameras to search dogs has been explored for several years and such a system was used at the World Trade Center. The concept does not compete with robots, as robots are used to enter places canines cannot. The canine team handlers have generally objected to dog-cams because the cameras and communications gear interfere with the dog's mobility, pose the hazard of snagging, and the dog cannot be readily commanded to stop at points of interest beyond the line-of-sight of the handler (dogs use visual cues as commands, more than audio). However, there is less objection to placing nodes in a rat's brain to stimulate and drive the rat into a void while carrying a camera or other sensors [50.35]. The motivation for a robot rat stems in part from the rat's mobility and relative low value. While the technical feasibility of a robot rat may be within reason, assuming advances in wireless communications, the response community has been lukewarm towards the idea [50.36]. Unlike rates, robots can be kept in storage for years and can pene-

trate through pockets of fire or areas with no oxygen. A robot rat has all of the limitations of dog, including the problem of a handler becoming too emotionally attached, and is likely to scare a trapped survivor just as much as the other rats that swarm a disaster. The con-

ventional wisdom is that, if the sensors, wireless, and power systems can be miniaturized and operate reliably enough to control a rat deep within rubble, those systems will enable responders and robots to work without the rats.

50.5 Evaluation and Benchmarks

There are a variety of methods for evaluating a rescue robot or the larger mixed human–robot system. *Simulations*, either through computer simulations or physical test beds duplicate some key component of a disaster or a disaster on a smaller scale, are common. Simulations provide availability and repeatability. *Demonstrations* at physical test beds, such as training sites for responders or through the RoboCup rescue competition, have been the primary source of feedback about rescue robots. Demonstrations can provide feedback on the performance of a robot under known conditions and to a lesser extent about the mixed-team performance. Demonstrations are often not repeatable but do offer a higher degree of physical fidelity with expected disaster settings than can be achieved through computer simulation. Simulations can be used for statistically significant experimentation, however, the need to control the number of variables in an experiment often leads to a form of reductionism, where a hypothesis is tested but under such artificial conditions as to undermine the value of the experiments. *Technology insertions* into actual disasters provide valuable information, though that information is not repeatable, and the ground truth is often unknown and generally relies on the skill of the observers.

In order to evaluate a rescue robot or mixed human–robot team, there must be metrics and measurement methods. Unfortunately, rescue robotics is such a new field of endeavor that metrics and methods are still currently under development. The scope of the evaluation influences the choice of metrics and methods. Rescue robots can be evaluated within at least four different categories:

1. the physical attributes of the robot, which can be further broken down by subsystem
2. the human operator performance
3. the performance of the humans and robots together as a mixed team
4. the impact of the robot team on the survivors or non-team members

At this early stage in the development of rescue robots, metrics and methods are being proposed for physical, operator, and team performance [50.37].

Metrics and methods are used by technologists to measure progress towards a useful product while standards represent that product's minimum capabilities and attributes and enables consumers (in this case, the response community) to purchase worthwhile systems. Standards rely on metrics and methods for measuring compliance, though these may sometimes be different from those used by researchers and developers. The US government National Institute of Standards and Technology (NIST) is leading the effort to standardize rescue robots for adoption by US responders. This effort is being conducted through ASTM and thus, even though it is largely driven by the US, will likely result in an international standard that may be adopted by other countries. NIST has considerable experience in testing rescue robotics, beginning with the creation in the late 1990s of a portable standard test bed for search and rescue for use by the RoboCup rescue and AAAI mobile robot competitions.

50.5.1 Computer Simulations for Rescue Robotics

Computer simulations provide a low-cost mechanism to explore the larger behavior of a robot or system. Generally, computer simulations provide high fidelity for testing software execution, but their physical fidelity depends on the physics engine. Simulating sensors and the complex environments produced by a disaster is difficult and is rarely accurate enough to test perceptual algorithms. At the time of writing, two readily available computer simulations exist for exploring the strategic and tactical applications of rescue robots within the RoboCup rescue framework [50.38], the *RoboCup rescue simulation project* [50.39] and *USARSim* [50.40]. These simulations are well understood, accepted, open-source, and free; as such, they should be useful for most

researchers or practitioners interested in ground-based rescue robotics.

The RoboCup rescue simulation project is used in the RoboCup rescue simulation league to study agent-based approaches to strategic planning for the disaster response. The simulator assumes a strong centralized response capability that is not necessarily the case for all countries or regions; the United States, for example, relies on a highly distributed organization that obviates many centralized coordination schemes. Although the simulation is focused on strategic decision making, particularly dynamic resource allocation, it does support the examination of how robot resources might be allocated during a disaster and how data from a robot might be propagated through a system. It permits the simulation of monitoring of disaster damage from reports by humans, distributed sensors, and robots and can simulate complex interactions such as telemedicine.

USARSim is a computer simulation developed by the University of Pittsburgh for physical robot simulation in disaster situations [50.40]. The simulation replicates the NIST standard test bed for search and rescue and permits efficient prototyping and testing of robot design and most aspects of control software. It uses Unreal game engine for handling physics and graphics, and virtual robots have capability of sensing (image, laser range finder, etc.) and actuation (wheel, motor, etc.) with data processing (image recognition, SLAM, etc.) in artificial environments. In 2006, the RoboCup rescue competition created a simulation league using this environment.

50.5.2 Physical Test Beds

Physical test beds provide a more realistic venue than a computer simulation for evaluating rescue robots, but may not be available to researchers, too expensive to use or to travel to, or not adequately capture some key aspect of a disaster. Physical test beds generally fall into two categories: *test beds developed for the fire rescue community* or the *NIST standard test bed for search and rescue*.

Fire rescue training test beds occur throughout the world and are used to train human firefighters and rescue specialists. Some facilities are also used to train canine teams. These test beds are constructed from construction and sewer debris, can introduce smoke and some simulants, and pose challenging mobility conditions, but vary in terms of fidelity. In many test beds, the density of the debris does not contain the actual amount of metals in a real collapse. This can lead to optimistic reports of

success of sensors and wireless communication devices. The test beds, being designed for human training, do not replicate the conditions under which a ground robot would be used. The terrain is generally on the exterior of the rubble and does not exercise the robot in confined



Fig. 50.9a–c View of the NIST standard test bed for search and rescue used by RoboCup Rescue. (a) View of overall test bed, (b) dummy representing a victim, and (c) a step field challenging robot mobility (courtesy NIST)

or vertical spaces. Depending on the size of the facility, the test bed may or may not be suitable for evaluating UAVs. An example of a fire training test bed appeared earlier in Fig. 50.7b.

Perhaps the most influential physical simulation for researchers is the RoboCup rescue physical league which uses the NIST standard test bed for search and rescue, shown in Fig. 50.9. This competition started in 2001 [50.41] and has more than 40 team entries every year from all over the world. The RoboCup rescue physical league scores robot performance in terms of mobility, mapping, situation awareness, sensing, shared autonomy, etc. A robot or robot team competes in one of three arenas which simulate disaster situations at the annual RoboCup world competition. The mission of the robot teams is to collect victim information (existence, state, and location) by sensor fusion of vital signals (heat, shape, color, motion, sound, CO₂, etc.) and report a map of victims in disaster space so that responders can efficiently arrive at the victim for rescue. In addition to the arenas, the competition and test bed contain individual skill test stations, for example, in order to test mobility, robots must traverse a random step field made of wood. The test bed was designed to be portable and reasonably inexpensive and several locations around the world have set up duplicates. As a result of the constraints of cost and portability, the test bed is not fully

representative of actual disaster physical conditions and does not test the operating conditions for the human teams.

50.5.3 Standards Activity

Standards for rescue robots and systems are being generated at the time of writing. The E54.08 subcommittee on operational equipment within the E54 Homeland Security application committee of ASTM International started developing an urban search and rescue (USAR) robot performance standard with the National Institute of Standards and Technology (NIST) as a US Department of Homeland Security (DHS) program from 2005 to 2010. It plans to cover sensing, mobility, navigation, planning, integration, and operator control in order to ensure that the robots can meet operational requirements under the extreme conditions of rescue. The standards will consist of performance measures that encompass basic functionality, adequacy and appropriateness for the task, interoperability, efficiency, and sustainability. The components of the robot systems include platforms, sensors, operator interfaces, software, computational models and analyses, communication, and information. Development of requirements, guidelines, performance metrics, test methods, certification, reassessment, and training procedures is planned.

50.6 Fundamental Problems and Open Issues

Rescue robotics is clearly a challenging field with many open issues. This section discusses the challenges by subsystem (mobility, communications, control, sensors, power), then the general human–robot interaction (HRI) challenges, and finally the remaining problems in evaluation. In addition to these specific areas discussed below, there are three cross-cutting challenges that should be mentioned: how to reduce mission times; how to localize, map, and integrate data from the robots into the larger geographic information systems used by strategic decision makers; and how to improve the overall reliability of rescue robots.

One of the most pressing challenges that cut across all subsystems is how to make rescue robot operations more efficient in order to find more survivors or provide more timely information to responders. Since rescue robots typically perform missions that humans cannot do, it is hard to compare performance directly. However, it may be useful to consider how robot operations can

be made more efficient in terms of pre-mission preparation (e.g., how long does it take to set up the robot system), mission execution, and post-mission activities (e.g., how long does it take to change batteries, decontaminate, inspect for wear, or perform minor repairs, etc.). Efficient mission execution remains the subject of much research, particularly investigations into how to speed up navigation through rubble or damaged areas with faster robots or autonomy. Studies have consistently shown that ground robots operating in the interior of collapsed buildings are physically moving only 51% of the time and that only 44% of the operator's activities are directly related to navigation [50.42]. Therefore speeding up mission activities such as building situation awareness and recognition may be as important to any type of rescue mission as speeding up navigation.

Rescue robots are tasked with missions where geographical information would help responders. For example, for ground robots, finding the presence of

a survivor in rubble is helpful, but knowing the location of the survivor is better, and knowing the path of the robot and the condition or type of rubble along the path even more helpful to extrication. Unfortunately, ground and surface vehicles are generally tasked in areas where structures prevent reliable GPS reception and underwater vehicles cannot receive GPS signals. Fixed-wing aerial vehicles, such as those used after hurricane Katrina, Rita, and Wilma, generally come with software that permits path mapping in GPS coordinates that can be displayed on maps, as well as determining the GPS location of any object in the camera view. Improvements to ground and surface vehicle mapping are expected to be a function of sensor miniaturization. As seen in Chap. 38, significant progress has been made in three-dimensional (3-D) simultaneous localization and mapping using range sensors larger than fieldable rescue robots.

Reliability of rescue robotics is obviously important, as robots which frequently break or need copious amounts of maintenance will be discarded. Failures may stem from physical failure, due to manufacture or unanticipated demands of the environment, or human error, including design failures or operator errors. As of 2004, ground rescue robots exhibited a mean time between physical failures (MTBF) of only 20 h, meaning that robots break every other shift, well below the desired 96 h proposed by the US military as the minimum for ground robots for urban operations [50.43]. In practice, a field ground or aerial robot system consists of two or three robots to provide complete redundancy, plus tools and spare parts.

50.6.1 Mobility

Mobility remains a major problem for all modalities of rescue robots, but especially for ground vehicles being used for urban search and rescue. The challenges for ground robots stem from the complexity of the environment, which is an unpredictable combination of vertical and horizontal elements with unknown surface characteristics and obstacles. The field is currently lacking any useful characterizations of rubble to facilitate better design. However, even without a complete understanding of rubble environments, it is clear that more work is needed in actuation and mechanical design as well as in algorithms that would enable the robot to adapt its mobility style to the current terrain (also known as *task shaping* [50.44]). Aerial vehicles, particularly helicopters, are vulnerable to wind conditions near (or in) structures and obstacles such as power lines, trees, and

overhanging debris. Surface and underwater vehicles have to contend with swift currents and floating or submerged debris, placing significant demands on agility and control.

50.6.2 Communications

Robots rely on real-time communications for teleoperation and for enabling responders to see what the robot is seeing immediately. Ground robots communicate either through a tether or via wireless radio. Aerial and surface robots are wireless, while underwater rescue robots are controlled via a tether. The communication bandwidth demands of all modalities are generally high due to the use of video imagery, and the tolerance to communications latency is low due to the control needs. In addition to communications between the tactical rescuers and their robots, it is difficult to report or transfer critical information provided by rescue robots to the strategic enterprise. Disasters typically destroy the communication infrastructure, both telephones and cellular phones, and alternatives such as satellite phones become saturated by response agencies.

Wireless communications with robots remain problematic. Operations below ground or near structures interfere with the physical propagation of radio signals. As shown in high-fidelity US&R response exercises, ad hoc wireless networks established by responders are likely to become quickly saturated, with no way of establishing priority over information. At the World Trade Center, data from the Solem robot deployed in the interior of building WTC4 returned totally black frames for 1 min 40 sec of the 7 min run before wireless communications were totally lost and the robot was abandoned [50.10]. In addition, many wireless robots use lossy compression algorithms to manage bandwidth, which interfere with computer vision techniques, and/or connect through insecure links, raising the possibility that news media might intercept and broadcast sensitive video of trapped victims.

Rescue robots working underground, either for US&R or mine rescue, have two alternatives to wireless communications: either operate with a tether or deploy repeaters to maintain wireless communications. Many wireless robots now can be purchased with a fiber-optic tether; however, these tethers are fragile and may break or tangle. The fiber-optic tether may also tangle with the safety rope used to support the robot during vertical drops. Data for the World Trade Center deployments indicated that a dedicated person was required to manage the tether but that 54% of tether management operations

were to allow the robot to reach a more favorable position or to recover the robot after its mission [50.10]. Hybrid communications, in which a robot is primarily on a tether then operates for short distances over a local wireless link before reconnecting to the tether, appear to be attractive.

50.6.3 Control

Robot control can be subdivided into *platform control*, which is usually considered by control theory, and *activity control*, which generally falls under the purview of artificial intelligence. Rescue robots are challenging both for traditional control and for artificial intelligence. The high degree of mechanical complexity of all modalities and the demands of the environment present major challenges for control theory. The robot's activity is usually handled by teleoperation; a human is needed to direct the robot and to perform mission sensing. The well-documented problems of manual teleoperation (see Chap. 32 Telerobotics) argue for increasing autonomy [50.45].

50.6.4 Sensors

Sensors, and sensing, pose the greatest mission challenge; without adequate sensing, a robot may be in an area of interest but be unable to navigate or to execute the larger mission. The physical attributes of a sensor (size, weight, and power demands) impact on whether it can be used with a particular robot platform. Currently sensors are not interchangeable between platforms; standards are needed for footprint sizes, mounting, connections, and display space.

The functionality of a sensor depends on the modality and mission [50.17]. The primary sensor missing from all robot modalities operating outside of water is a miniature range sensor. With a miniature range sensor, the success in localization and mapping seen with larger robots would be transferable to rescue robots. The sensor payloads for other missions depend on those applications; however, two sensor needs for US&R are particularly noteworthy. One is a detector that can perceive victims obscured by rubble; current radars have not been reliable in mixed rubble. Another needed sensor is one that can tell if a victim is unconscious but alive without touching the victim. A robot may be able to see a victim, yet not be able to crawl to the victim and make contact or scrape away enough dirt or clothing to take a pulse. Stand-off detectors such as millimeter-wave

radars and gas detectors appear promising but have not been validated at this time.

Smaller, better sensors are not sufficient; improvements in sensing algorithms are also needed. At this time, humans are expected to interpret all sensing data manually in real time. This is a daunting task for many reasons. Human performance is handicapped by physiological factors introduced by sensing through a computer display (also known as computer mediation), the location of sensors at viewpoints low to the ground, generally restricted fields of view, and fatigue. The modality output itself may also be nonintuitive, such as ground-penetrating radar. However, autonomous detection and general scene interpretation is considered well beyond the capabilities of computer vision. This presents a case where neither the human nor the computer can accomplish the perceptual task reliably and argues for investigation into human-computer cooperative techniques for perception. Algorithms that enhance the image for human inspection, supplement depth perception, or cue interesting areas are within the reach of computer vision.

50.6.5 Power

The robot modality and mission poses distinct challenges for power. In general, battery power is preferred over internal combustion because of the logistics difficulties in transporting flammable liquids. While the requirements of each rescue robot application is largely unknown, a partial understanding is emerging of the power profile. For example, the operation tempo of a ground vehicle operating underground is on the order of 3–4 runs, each around 20 min in duration, over a 12–14 h shift, with the robot kept on hot standby for the majority of the shift. Rotary-wing aerial vehicles for tactical reconnaissance and structural inspection show an operations tempo of 5–8 min per face of a building, while fixed-wing vehicles are airborne for less than 20 min. Other rescue missions, such as wilderness search and rescue, will have different requirements but the need for batteries over internal combustion and for determining the power profile is the same.

The size and weight of the power source is important. It may be the single driver in the overall size of the platform and influence key design attributes such as the placement of the payload. A word of warning is warranted: a common design flaw in rescue robots is to locate the batteries so as to maximize stability but with the unintended consequence of making it very time-

consuming to reach and replace them in the field without special tools.

50.6.6 Human–Robot Interaction

Human–robot interaction (HRI) has emerged as a major challenge in rescue robotics and has been declared to be an exemplar domain within HRI [50.12]. Robots are part of a humancentric system; even if robots were fully autonomous, responders want the information in real time and the option to modify the robot’s tasks. There are at least four key problems to be resolved in HRI. First, there is the issue of *human-to-robot ratio* for operations. Currently a single robot requires between two and three humans, depending on modality, for safe reliable operation. Second, these *operators have to be extensively trained*, which may be out of the question for many response teams. Third, user interfaces for operations do not provide sufficient situation awareness and need improving, which may in turn reduce training demands. Fourth, humans will interact with the robot outside of the operator role and this leads to a need for *affective robotics*. For example, a survivor will have a relationship with the *front* of a robot [50.46] as well as a rescuer working alongside a robot [50.47].

Reducing the human-to-robot ratio is probably the best known challenge for HRI. Unmanned aerial and water-based vehicles for defense and research applications have a long tradition of at least two operators per vehicle, one as a pilot, the other as a payload specialist. The number may increase due to payload or mission complexity or safety (e.g., the requirement for a safety officer). Unmanned ground vehicles have presumed that a single operator is sufficient to drive and look, though studies with ground rescue robots dispute this and show a ninefold increase in performance when two operators work cooperatively [50.42]. Since rescue workers work in teams of two or more, a single-operator single-robot control regime is not essential [50.12]. The need for multiple humans impacts on the viability of alternative scenarios, such as swarms or teams of multiple robots; these may require too many operators unless higher degrees of autonomy and situation awareness are possible.

Training is a related issue in human–robot interaction [50.48]. Rescue workers have limited time compared with military operations to learn about robots and few opportunities to practice. While a bomb squad or special weapons and tactics team for law enforcement may be called out several times each month, a rescue team may be called out only a few times each year. For

the near future, rescue workers may not have had prior training or experience with a robot prior to the disaster and be expected to use prototypes with only *hasty* training.

Situation awareness is defined by Endsley in [50.49] as “the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future.” Drury et al. [50.50] have defined types of situation awareness within search and rescue based on an analysis of RoboCup rescue tasks, while Casper and Murphy [50.51] and Burke et al. [50.42] have examined operator situation awareness in technology insertions. User interfaces are a key component in facilitating situation awareness. In rescue robotics, user interfaces are generally primitive and work through the operator’s visual channel to provide robot, task, and situation awareness of the state of the robot, task progress, and the general operational environment. To highlight the importance of user interfaces, one robot at the World Trade Center was rejected because of the complexity of its interface [50.10]. While the user interface for fieldable rescue robots primarily display the video output from the robot, experiences from RoboCup rescue suggests that a good interface will both facilitate commanding the robot (inputs) and will provide three types of information (outputs):

- the robot’s perspective: camera view(s) from the robot’s current position, plus any environmental perceptions that enhance the general impression of telepresence
- sensor and status information: critical information about the robots internal state and its external sensors
- if possible, a map: a bird’s eye view of the robot situated in the local environment

There has been a move towards videogame-like interfaces for ground vehicles; this is consistent with the 10 min *rule* of human–computer interaction that states that if a user cannot figure out how to use the main function of a computer within 10 min, the interface is flawed. However, the videogame interface may facilitate rapid, superficial control of the robot at the cost of limiting the transition to expert control which might not be best handled by a handheld controller. In general, all assessments of rescue robot interfaces agree that the operator’s visual channel is overloaded: the operator is expected to see too much and is likely to miss important information or become quickly fatigued. The use of other modalities such as sound, spoken language, and tactile feedback is warranted, though care must be taken

to consider whether such modalities will work in the noisy environs of a rubble pile or will interfere with personal protection equipment.

The focus of human–robot interaction on the operator and human–robot ratio has led to the issues associated with humans *in front* or *beside* the robot to be ignored. Ideally, rescue robots will interact with trapped survivors and facilitate their care and comfort. Likewise, a robot may be inserted into a confined space to act as the surrogate for a team member, such as a team leader or safety officer. Studies show that people view these robots socially, that is, they may ascribe traits of being scary or unhelpful based on the way the robot looks and moves [50.46]. Therefore, affective robotics applies to rescue robots as well as entertainment robots.

50.6.7 Evaluation

Evaluation of rescue robots is difficult not only because of the diversity of platforms and missions but also because each disaster is truly different. In addition, robots are part of a humancentric system: they are operated by humans in order to provide information to humans. Evaluation of the performance of a rescue robot system at an actual disaster is currently ad hoc. No computer

or physical simulation for predicting the performance of robots and humans in a disaster has been validated; indeed, there is little argument that simulations are far easier than a real response. The difficulties of simulation are exacerbated by the differences between disasters. For example, the World Trade Center was unique in terms of the large amount of steel and the density of the collapsed material, while earthquakes and hurricanes are different from terrorist events.

Metrics for measuring performance remain a worthwhile quest. Quantitative metrics, such as the number of survivors or remains found, do not capture the value of a robot in establishing that there are no survivors in a particular area. Performance metrics from psychology and industrial engineering are only now beginning to be applied. These methods require enhanced computer and full-scale simulations in order to collect data. Data collection on human and overall system performance during a disaster has been done through ethnographic observations and are now moving to direct observations of situation awareness during demonstrations [50.42, 52]. Direct data collection during a disaster may not be possible as methods may interfere with performance (and therefore be unreasonable, if not unethical) and arouse fears by operators of *Big Brother* and being held liable for any errors in operation.

50.7 Conclusions and Further Reading

Rescue robots are making the transition from an interesting idea to an integral part of emergency response. Aerial and ground robots have captured most of the attention, especially for disaster response, but water-based vehicles (both surface and underwater) are proving useful as well. Rescue robots present challenges in all major subsystems (mobility, communications, control, sensors, and power) as well as in human–robot interaction. Man-portable and man-packable systems are the most popular because of their reduced logistics burden, but the size of the platforms exacerbates the need

for miniaturized sensors and processors. Wireless communications remains a major problem. While recent deployments have relied on polymorphic tracked vehicles, researchers are investigating miniature planes and helicopters, new robot designs, especially biomimetic, and alternative concepts of operations. Standards are currently under development and this will help accelerate the adoption of rescue robots. The annual [IEEE Workshop on Safety, Security and Rescue Robotics](#) is currently the primary conference and clearinghouse for research in rescue robotics.

References

- 50.1 A. Davids: Urban search and rescue robots: from tragedy to technology, *Intell. Syst. IEEE* **17**(2), 81–83, 1541–1672 (2002) [see also *IEEE Intelligent Systems and Their Applications*]
- 50.2 J. Walter, International Federation of Red Cross and Red Crescent Societies: World disasters report 2005. (Kumarian Press, Bloomfield 2005)

- 50.3 *Standard on Operations and Training for Technical Rescue Incidents*. (National Fire Protection Association 1999)
- 50.4 *Technical Rescue Program Development Manual*. (United States Fire Administration 1996)
- 50.5 J.A. Barbera, C. DeAtley, A.G. Macintyre: Medical aspects of urban search and rescue, *Fire Eng.* **148**, 88–92 (1995)
- 50.6 R.R. Murphy, S. Stover: Gaps analysis for rescue robots. In: *ANS 2006: Sharing Solutions for Emergencies and Hazardous Environments* (American Nuclear Society, LaGrange Park 2006)
- 50.7 C. Schlenoff, E. Messina: A robot ontology for urban search and rescue. In: *ACM workshop on Research in knowledge representation for autonomous systems*, Bremen (Association for Computing Machinery, New York 2005) pp. 27–34
- 50.8 R. Murphy, S. Stover, H. Choset: Lessons learned on the uses of unmanned vehicles from the 2004 florida hurricane season. In: *AUVSI Unmanned Systems North America*, Baltimore (Association for Unmanned Vehicle Systems International, Arlington 2005)
- 50.9 J. Casper, R. Murphy: Human-robot interaction during the robot-assisted urban search and rescue effort at the world trade center, *IEEE Trans. Syst. Man Cybernet. B* **33**(3), 367–385 (2003)
- 50.10 R.R. Murphy: Trial by fire, *IEEE Robot. Autom. Mag.* **11**(3), 50–61 (2004)
- 50.11 S. Tadokoro, T. Takamori, S. Tsurutani, K. Osuka: On robotic rescue facilities for disastrous earthquakes – from the great hanshin-awaji (kobe) earthquake, *J. Robot. Mechatron.* **9**(1), 10 (1997)
- 50.12 R. Murphy: Human-robot interaction in rescue robotics, *IEEE Trans. Syst. Man Cybernet. Appl. Rev.* **34**(2), 138–153 (2004)
- 50.13 C. Manzi, M. Powers, K. Zetterlund: Critical information flows in the alfred p. murrah building bombing. Technical report, Chemical and Biological Arms Control Institute (2002)
- 50.14 F. Matsuno, S. Tadokoro: *Rescue Robots and Systems in Japan*. IEEE International Conference on Robotics and Biomimetics (2004) pp. 12–20
- 50.15 R. Murphy, E. Steimle, C. Cullins, K. Pratt, C. Griffin: Cooperative damage inspection with unmanned surface vehicle and micro aerial vehicle at hurricane wilma. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (video proceedings)*, Beijing (IEEE Press 2006)
- 50.16 R.R. Murphy, C. Griffin, S. Stover, K. Pratt: Use of micro air vehicles at hurricane katrina. In: *IEEE Workshop on Safety Security Rescue Robots*, Gaithersburg (IEEE Press 2006)
- 50.17 R. Murphy, J. Casper, J. Hyams, M. Micire, B. Minten: Mobility and sensing demands in usar. In: *IECON: Session on Rescue Engineering*, Vol. 1, Nagoya (IEEE Press 2000) pp.138–142
- 50.18 R.R. Murphy, S. Stover: Rescue robot performance at 2005 la conchita mudslides. In: *ANS 2006: Sharing Solutions for Emergencies and Hazardous Environments* (American Nuclear Society, LaGrange Park 2006)
- 50.19 K. Pratt, R.R. Murphy, S. Stover, C. Griffin: Requirements for semi-autonomous flight in miniature uavs for structural inspection. In: *AUVSI Unmanned Systems North America*, Orlando (Association for Unmanned Vehicle Systems International, Arlington 2006)
- 50.20 A. Wolf, H.B. Brown, R. Casciola, A. Costa, M. Schwerin, E. Shamas, H. Choset: *A Mobile Hyper Redundant Mechanism for Search and Rescue Tasks*. *International Conference on Intelligent Robots and Systems, 2003. (IROS 2003)*. *Proc. IEEE/RSJ*, Vol. 3 (2003) pp. 2889–2895
- 50.21 I. Erkmen, A. Erkmen, F. Matsuno, R. Chatterjee, T. Kamegawa: Snake robots to the rescue!, *IEEE Robot. Autom. Mag.* **9**(3), 17–25 (2002)
- 50.22 S. Hirose, E. Fukushima: Development of mobile robots for rescue operations, *Adv. Robot.* **16**(6), 509–512 (2002)
- 50.23 D. Campbell, M. Buehler: *Stair Descent in the Simple Hexapod 'RHex'*. *Conference on Robotics and Automation 2003. Proc. ICRA'03*, Vol.1 (2003) pp.1380–1385
- 50.24 R.M. Voyles, A.C. Larson: Terminatorbot: a novel robot with dual-use mechanism for locomotion and manipulation, *Mechatron. IEEE/ASME Trans.* **10**(1), 17–25 (2005)
- 50.25 W.E. Green, P.Y. Oh: A fixed-wing aircraft for hovering in caves, tunnels, and buildings. In: *American Control Conference* (IEEE Press 2006) pp. 1–6
- 50.26 N. Sato, F. Matsuno, T. Yamasaki, T. Kamegawa, N. Shiroma, H. Igarashi: *Cooperative Task Execution by a Multiple Robot Team and its Operators in Search and Rescue Operations*. *International Conference on Intelligent Robots and Systems, 2004. (IROS 2004)*. *Proc. IEEE/RSJ*, Vol.2 (2004) pp.1083–1088
- 50.27 R. Murphy: Marsupial and shape-shifting robots for urban search and rescue, *IEEE Intell. Syst.* **15**(3), 14–19 (2000)
- 50.28 D.P. Stormont, A. Bhatt, B. Boldt, S. Skousen, M.D. Berkemeier: *Building Better Swarms Through Competition: Lessons Learned from the Aaii/Robocup Rescue Robot Competition*. *International Conference on Intelligent Robots and Systems, 2003. (IROS 2003)*. *Proc. IEEE/RSJ*, Vol. 3 (2003) pp. 2870–2875
- 50.29 R. Murphy: Biomimetic search for urban search and rescue. In: *IROS 2000*, Vol. 3. Takamatsu (2000) pp. 2073–2078
- 50.30 V. Kumar, D. Rus, S. Singh: Robot and sensor networks for first responders, *IEEE Pervasive Comput.* **3**(4), 24–33 (2004)

- 50.31 D. Kurabayashi, H. Tsuchiya, I. Fujiwara, H. Asama, K. Kawabata: *Motion Algorithm for Autonomous Rescue Agents Based on Information Assistance System*. *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, 2003, Vol. 3 (2003) pp. 1132–1137
- 50.32 K.W. Sevcik, W.E. Green, P.Y. Oh: Exploring search-and-rescue in near-earth environments for aerial robots. In: *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)* (IEEE Press, 2005) pp. 693–698
- 50.33 J. Tanaka, K. Suzumori, M. Takata, T. Kanda, M. Mori: *A Mobile Jack Robot for Rescue Operation. Safety, Security and Rescue Roboticsm Workshop*, 2005. *Okayama Uni.* (IEEE International, 2005) pp. 99–104
- 50.34 R. Murphy, T. Vestgaarden, H. Huang, S. Saigal: *Smart Lift/Shore Agents for Adaptive Shoring of Collapse Structures: A Feasibility Study*. *IEEE Workshop on Safety Security Rescue Robots* (Gaithersburg 2006)
- 50.35 L. Yihan, S.S. Panwar, S. Burugupalli: *A mobile sensor network using autonomously controlled animals*. *Proceedings of the First International Conference on Broadband Networks (BROAD-NETS'04)* (IEEE Computer Society Press, 2004) pp. 742–744
- 50.36 R. Murphy: Rats, robots, and rescue, *IEEE Intell. Syst.* **17**(5), 7–9 (2002)
- 50.37 T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of socially interactive robots, *Robot. Autonom. Syst.* **42**(3–4), 143–166 (2003)
- 50.38 S. Tadokoro, H. Kitano, T. Takahashi, I. Noda, H. Matsubara, A. Hinjoh, T. Koto, I. Takeuchi, H. Takahashi, F. Matsuno, M. Hatayama, J. Nobe, S. Shimada: The robocup-rescue project: a robotic approach to the disaster mitigation problem. In: *IEEE International Conference on Robotics and Automation*, Vol. 4 (IEEE Press, 2000) pp. 4089–4094
- 50.39 T. Takahashi, S. Tadokoro: Working with robots in disasters, *IEEE Robot. Automat. Mag.* **9**(3), 34–39 (2002)
- 50.40 I.R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, S. Burion: Human-robot teaming for search and rescue, *IEEE Pervasive Comput.* **4**(1), 72–79 (2005)
- 50.41 H. Kitano, S. Tadokoro: Robocup-rescue: A grand challenge for multi-agent and intelligent systems, *AI Mag.* **22**(1), 39–52 (2001)
- 50.42 J. Burke, R. Murphy, M. Covert, D. Riddle: Moonlight in miami: An ethnographic study of human-robot interaction in usar, *Human-Comput. Interact.* **19**(1–2), 85–116 (2004), special issue on Human-Robot Interaction
- 50.43 J. Carlson, R. Murphy: How ugvs physically fail in the field, *IEEE Trans. Robot.* **21**(3), 423–437 (2005)
- 50.44 G.M. Kulali, M. Gevher, A.M. Erkmen, I. Erkmen: *Intelligent Gait Synthesizer for Serpentine Robots*. *IEEE International Conference on Robotics and Automation*, 2002. *Proc. ICRA'02*, Vol. 2 (2002) pp. 1513–1518
- 50.45 A. Birk, S. Carpin: Rescue robotics: a crucial milestone on the road to autonomous systems, *Adv. Robot.* **20**(5), 596–605 (2006), 595
- 50.46 R. Murphy, D. Riddle, E. Rasmussen: Robot-assisted medical reachback: a survey of how medical personnel expect to interact with rescue robots. In: *13th IEEE Int. Workshop on Robot and Human Interactive Communication, ROMAN* (IEEE Press, 2004) pp. 301–306
- 50.47 T. Fincannon, L.E. Barnes, R.R. Murphy, D.L. Riddle: *Evidence of the Need for Social Intelligence in Rescue Robots*, Vol. 2 (2004) pp. 1089–1095
- 50.48 R. Murphy, J. Burke, S. Stover: *Field Studies of Safety Security Rescue Technologies Through Training and Response Activities*. *International Conference on Intelligent Robots and Systems*, 2004 (*IROS 2004*). *Proc IEEE/RSJ*, Vol. 2 (2004) pp. 1089–1095
- 50.49 M. Endsley: Design and evaluation for situation awareness enhancement. In: *Human Factors Society 32nd Annual Meeting* (Santa Monica, CA 1988) pp. 97–101
- 50.50 J.L. Drury, J. Scholtz, H.A. Yanco: *Awareness in Human-Robot Interactions*. *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1 (2003) pp. 912–918
- 50.51 J. Casper, R. Murphy: Workflow study on human-robot interaction in usar, *ICRA* **2002**, 1997–2003 (2002)
- 50.52 J. Scholtz, B. Antonishek, J. Young: *A Field Study of Two Techniques for Situation Awareness for Robot Navigation in Urban Search and Rescue*. *IEEE International Workshop on Robot and Human Interactive Communication*, 2005. *ROMAN 2005* (2005) pp. 131–136

Intelligent Vehicles

51. Intelligent Vehicles

Alberto Broggi, Alexander Zelinsky, Michel Parent, Charles E. Thorpe

This chapter describes the emerging robotics application field of intelligent vehicles – motor vehicles that have autonomous functions and capabilities. The chapter is organized as follows. Section 51.1 provides a motivation of why the development of intelligent vehicles is important, a brief history of the field, and the potential benefits of the technology. Section 51.2 describes the enabling technologies for intelligent vehicles to sense vehicle, environment and driver state, work with digital maps and satellite navigation, and communicate with intelligent transportation infrastructure. Section 51.3 describes the challenges and solutions associated with road scene understanding – a key capability for all intelligent vehicles. Section 51.4 describes advanced driver assistance systems, which use robotics and sensing technologies described earlier to create new safety and convenience systems for motor vehicles, such as collision avoidance, lane keeping, and parking assistance. Section 51.5 describes driver monitoring technologies that are being developed to mitigate driver fatigue, inattention, and impairment. Section 51.6 describes fully autonomous intelligent vehicles systems that have been developed and deployed. The Chapter is concluded in Sect. 51.7 with a discussion of future prospects, while Sect. 51.8 provides references to further reading and additional resources.

51.1	Why Intelligent Vehicles?	1175
51.1.1	Brief History	1176
51.1.2	Benefits of Intelligent Vehicles.....	1177
51.2	Enabling Technologies	1178
51.2.1	Environment State	1179
51.2.2	Driver State	1180
51.2.3	Communication	1180
51.2.4	Digital Maps and Satellite Data	1180
51.3	Road Scene Understanding	1181
51.3.1	Road/Lane Tracking	1182
51.3.2	Road Sign Detection	1182
51.3.3	Traffic-Light Detection	1183
51.3.4	Visibility Assessment.....	1183
51.3.5	Vehicle Detection	1183
51.3.6	Pedestrian Detection	1184
51.4	Advanced Driver Assistance	1184
51.4.1	Collision Avoidance and Mitigation.....	1185
51.4.2	Adaptive Cruise Control	1187
51.4.3	Stop and Go	1187
51.4.4	Parking Assist	1187
51.4.5	Lane Keeping.....	1188
51.4.6	Lane Changing.....	1188
51.5	Driver Monitoring	1188
51.5.1	Driver Fatigue, Inattention, and Impairment	1189
51.5.2	Driver and Passenger Protection.....	1190
51.5.3	Emergency Assistance	1190
51.6	Automated Vehicles	1191
51.6.1	Operating Safely.....	1191
51.6.2	Traffic Congestion.....	1191
51.6.3	Environmental Factors	1191
51.6.4	The Automobile of the Future	1192
51.6.5	Automated Vehicle Deployment	1193
51.7	Future Trends and Prospects	1194
51.8	Conclusions and Further Reading	1195
	References	1195

51.1 Why Intelligent Vehicles?

An important field of application of robotics has emerged in the last 20–25 years which is centered on

the automobile, named *intelligent vehicles*. The automobile has been one of the most important products of

the 20th century. It has generated an enormous industry and has given individuals a freedom of movement that has completely changed our ways of living. Indeed, the automobile has been a key factor in the large change in the way our urban societies are structured. Today, there are more than 800 million vehicles on the planet and this number is expected to double in the next ten years. This challenge has led to the development of an active research domain with the ultimate goal of automating the typical tasks that humans perform while driving. An intelligent vehicle is defined as a vehicle enhanced with perception, reasoning, and actuating devices that enable the automation of driving tasks such as safe lane following, obstacle avoidance, overtaking slower traffic, following the vehicle ahead, assessing and avoiding dangerous situations, and determining the route. The overall motivation of building intelligent vehicles has been to make motoring safer, and more convenient and efficient.

51.1.1 Brief History

The history of intelligent vehicles has developed over the last two decades. Although the first ideas were born in the 1960s, the level of maturity of the technology at that time did not allow pursuit of the original goal of implementing fully autonomous all-terrain all-weather vehicles. The first documented prototypes of automated vehicles were fielded by a few groups in the military arena in the mid 1980s [51.1–3]. The initial stimulus that triggered these innovative ideas was provided by the military sector, which was eager to provide complete automation to its fleet of ground vehicles.

It was not before the 1980s that this interest was transferred to the civil sector: governments worldwide launched the first projects, which supported a large number of researchers in these topics. The interest of the automotive industry in developing real products was only triggered after feasibility studies were successfully completed and the first prototypes were demonstrated. Testing of autonomous vehicles on real roads in a real environment was one of the most important milestones in the history of intelligent vehicles. This happened in the mid to late 1990s. Figure 51.1 shows the first motor vehicles that pioneered the development of intelligent vehicles. In the summer of 1995, the Carnegie Mellon Navlab group ran their *No Hands Across America* experiment [51.4]. They demonstrated automated steering, based solely on computer vision, over 98% of the time on a 2800 mile trip across the United States. Later in 1995 the Bundeswehr Universität Munich (UBM), Germany fielded a vehicle that was demonstrated with a 1758 km

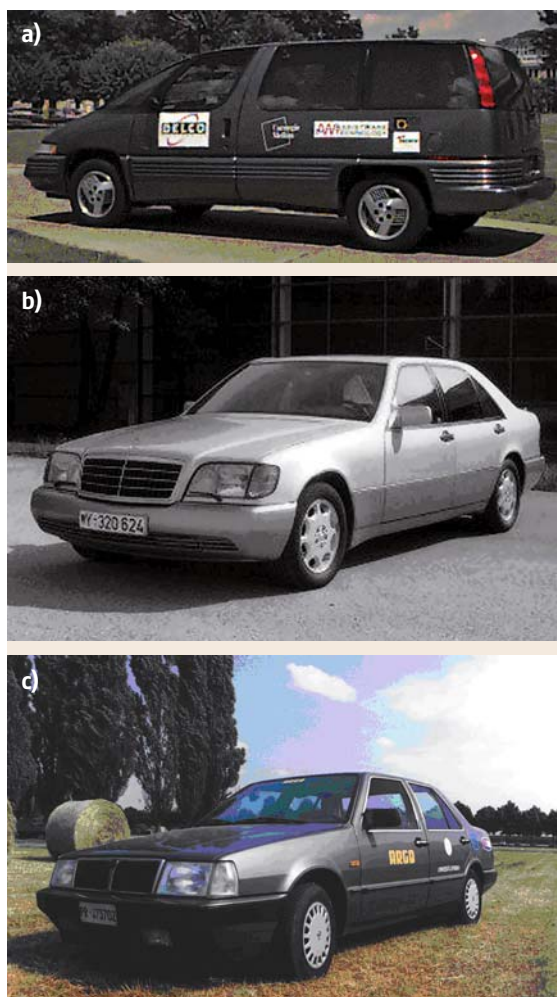


Fig. 51.1a–c Pictures of pioneering autonomous vehicles, from *top to bottom*: (a) NAVLAB, (b) UBM, and (c) Argo vehicles

trip from Munich to Copenhagen in Denmark and back. The vehicle was able to drive autonomously for 95% of the trip. The car suggested and executed maneuvers to pass other cars. Unlike later robot cars, this car located itself on the current road and followed it until instructed otherwise. It did not localize itself in global coordinates and could drive without Global Positioning System (GPS) and road maps as found in a modern automotive navigation systems. The car's trunk was full of transputers and ad hoc hardware. A different approach was followed by VisLab at the University of Parma within the Argo project: the passenger car that was designed and developed was based on a low-cost approach. An

off-the-shelf Pentium 200 MHz personal computer (PC) was used to process stereo images obtained from low-cost cameras installed in the driving cabin. The vehicle was able to follow the lane, locate obstacles, and – when instructed – change lane and overtake slower vehicles. The main milestone of this project was the successful test of the Argo vehicle in a tour of Italy of more than 2000 km called ‘Millemiglia in Automatico’ in which the vehicle drove itself for 94% of the total distance.

Current research initiatives are oriented towards the development of intelligent vehicles in realistic scenarios. However, due mainly to legal issues, full autonomy has not yet been set as the ultimate goal: the automotive industry has set as its primary goal the need to equip vehicles with supervised systems and – more generally – advanced driving assistance systems (ADAS) instead of automatic pilots. In other words, the driver is still in charge of running the vehicle, but the drive is monitored by an electronic system that detects possibly dangerous situations and reacts by either warning the driver in due time, or taking control of the vehicle in order to mitigate the consequences of the driver’s inattention. Given the aforementioned legal issues, the ultimate goal has shifted towards driving assistance systems since com-

plete vehicle automation was not felt to be a primary strategic area of investment for automotive industries. Concurrently, Departments of Transportation worldwide have been primarily interested in social, economic, or environmental objectives aimed at enhancing fuel efficiency, road network usage, and improving quality of life in terms of mobility [51.5].

The good results obtained by ADAS in the automotive arena in recent years has induced the military sector to give a new vigorous push to the original ideas of automating its fleet of ground vehicles. The Defense Advanced Research Projects Agency (DARPA) launched the Grand Challenge in 2003, a race for autonomous vehicles that had to travel for more than 200 km in unstructured environments. This unprecedented challenge attracted a large number of top-level research institutes, who worked with the million-dollar prize in mind and helped the scientific community take a considerable step forward.

In 2005 the DARPA Grand Challenge required autonomous driving in a rough terrain desert scenario with no traffic, obstacle types known in advance, and few if any road markers on a course predefined by 2935 GPS points. Five cars (maximum speed 40 km/h) completed the 211 km desert course:

1. Volkswagen of Stanford (Stanley 6 h 54 min)
2. Hummer of CMU (Sandstorm 2.5% slower)
3. CMU’s second Hummer (Highlander 5% slower)
4. Gray team (Kat 5 8% slower)
5. Terramax truck (44% slower)

Figure 51.2a–e shows photographs of the five finishing vehicles. Refer to [51.6] for the technical details of the DARPA Grand Challenge.

51.1.2 Benefits of Intelligent Vehicles

Having intelligent vehicles running on our road network would bring a number of social, environmental, and economical benefits. An intelligent vehicle able to assess the driving scenario and react in case of danger would allow up to 90% of traffic accidents that are caused by human errors to be eliminated, saving human lives. According to the World Health Organization an estimated 1.2 million people worldwide are killed each year, and about forty times this number injured, due to traffic accidents.

At the same time, vehicles able to drive at high speeds and very close to each other would decrease fuel consumption and polluting emissions; furthermore they would also increase road network capacity. Vehicles communicating with a ground station could share their



Fig. 51.2a–e Grand Challenge Vehicles (a) Stanley (1st) (b) Sandstorm (2nd), (c) Highlander (3rd), (d) Kat-5 (4th) and (e) Terramax (5th)

routes and be instructed to reroute in order to maintain a smooth traffic flow. Vehicles that can sense and obey speed limits or traffic rules would reduce the possibility of misinterpretation and antisocial driving behavior.

Fully automatic vehicles would also offer a higher degree and quality of mobility to a larger population, including young, old, or infirm individuals, reducing the need even for a driving licence. Finally, the availability of vehicles that could drive themselves would increase the quality of mobility for everyone, turning personal vehicles into taxis able to pick up people and take them to their final destination in total safety and comfort, dedicating the driving time to their preferred activities.

However, this full application of intelligent vehicles is far from being complete, since unmanned vehicles technology is still under development for many other applications. The automation of road vehicles is perhaps the most common everyday task that attracts the

greatest interest from the industry. However, other domains such as agricultural, mining, construction, search and rescue, and other dangerous applications in general, are looking to autonomous vehicles as a possible solution to the issue of the ever-increasing cost of personnel. If a vehicle could move autonomously on a field to seed, or enter a mined field, or even perform dangerous missions, the number of individuals put at risk would drastically decrease and at the same time the efficiency of the vehicle itself would be increased thanks to a 24/7 operational schedule. The key challenge for intelligent vehicles is safety; accidents must not occur due to automation errors and there is zero tolerance to human injury and death.

This chapter focuses on road and traffic applications of intelligent vehicles, which are catalyzing the interest of the automotive industry, car makers, and providers of automotive technologies.

51.2 Enabling Technologies

The basic sensing and actuation technologies for intelligent vehicles are readily available on the market. The key challenge to integrating new technologies is heavily dependent on the control strategies associated with the sensor data processing and reasoning. Automated systems must consider all subsystems in a vehicle, including the interaction of the technology with the driver. Key drivers for the new technologies in an intelligent vehicle are the desired applications. Solutions are developed on the premise that the system must satisfy the requirements of the application with the minimum level of technology.

Intelligent vehicle applications require the following:

- The position, and kinematic and dynamic state of the vehicle
- The state of the environment surrounding the vehicle
- The state of the driver and occupants
- Communication with roadside infrastructure or other vehicles
- Access to digital maps and satellite data

Position localization is a key technology for intelligent vehicles. The position of the vehicle must be known if the vehicle is to be controlled along a particular trajectory. To control an intelligent vehicle to perform applications such as collision avoidance or automatic lane changing requires knowledge of the kinematic and

dynamic states of the vehicle. Standard robotics techniques for position, kinematics, and dynamics are used in intelligent vehicle applications (see Chap. 34). Encoders are mounted on the steering column and the tail shaft. Vehicle heading, speed, and acceleration can be computed from the encoders using standard techniques (see Chap. 20).

To execute the trajectories in an intelligent vehicle a number of techniques are used. In many current applications of fully automated vehicles, the trajectories are fixed with limited choices at intersections. To execute the trajectories, the vehicles simply follow markers such as an antenna (a wire carrying an alternating current), magnets or passive transponders such as radio frequency identification (RFID) tags that are imbedded in the road or simple painted lines. In more advanced navigation systems, the vehicle generates trajectories in a map stored in its memory and localizes itself with respect to this map in order to execute them. This localization can be done using absolute positioning through a combination of global navigation satellite systems (GNSS), vehicular kinematic state, and inertial navigation. Alternatively, position localization can be done by relative localization using local markers such visual landmarks. (For specific reference to earlier mobile robot landmark navigation see Chap. 36.)

In order for the vehicle to execute trajectories, it needs to control the speed and the steering of the

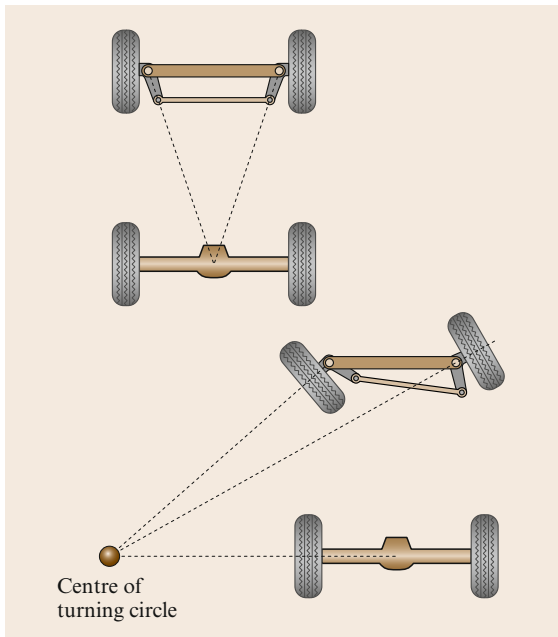


Fig. 51.3 Ackerman steering model

vehicle. Modern vehicles now incorporate electronic control units to control the acceleration (motor torque and possibly gearbox and clutch), the braking (electric or electrohydraulic brake control) and the steering angle (electric power assistance), and it is therefore quite easy

to implement trajectory control of the vehicles [51.7]. The safety of these critical functions remains a challenge, and is usually solved through redundancies in the sensors, actuators, and control units.

51.2.1 Environment State

Sensing the state of the environment surrounding the vehicle is a critical aspect of intelligent vehicle applications. The most difficult function for intelligent vehicles is road scene understanding. This includes locating key landmarks: the road, other vehicles, pedestrians, traffic signals, road signs, and other unstructured obstacles. A more difficult challenge is speed control following the detection of event in the road scene. The common sensors are infrared [51.8], ultrasound [51.9], radar [51.10], laser range finders [51.11], and computer vision [51.12], which continually scan the environment as shown in Fig. 51.4. Radar is generally used for obstacle detection at a distance, while infrared and ultrasound are used for close proximity obstacle detection. Laser ranging and image processing are used to more robustly recognize the road scene under various weather conditions. Certain road scene conditions such as road signs and traffic lights can only be understood using vision sensing. Sensor fusion is commonly used in intelligent vehicle applications, particularly between monocular vision and radar/laser sensors. This work is principally being done by the tier 1 automotive suppliers (Bosch, Denso, Del-

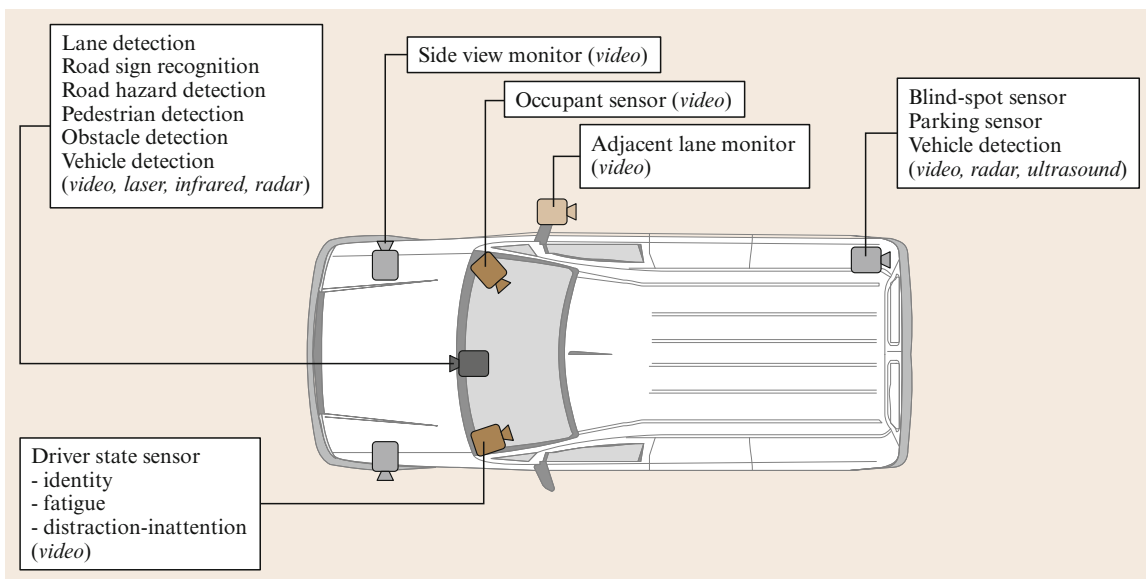


Fig. 51.4 Environment state sensing

phi, Visteon, Siemens, NEC) in cooperation with the major automotive manufacturers.

Using such sensors it is possible to map the environment surrounding the vehicle and then, using techniques such as simultaneous localization and mapping (SLAM) (Chap. 37), generate the complex manoeuvres needed for parking or for obstacle avoidance.

51.2.2 Driver State

Additionally an intelligent vehicle needs to understand the driver state, if it is to give appropriate warnings or need to take action. Vision sensors [51.13] can monitor a driver's attentiveness and fatigue by observing the direction of the driver's gaze and eyelid behavior. In an emergency knowing the position of the driver's head can assist in the safe deployment of airbags. Also, after an accident, observing the state of the driver and other occupants could be useful for the dispatch of emergency services to the accident scene.

51.2.3 Communication

Communication technologies enable interesting intelligent vehicles applications. Allowing vehicles to communicate with each other and with the highway offers the possible to create vast improvements in the safety and efficiency of the road system. Applications include intersection collision avoidance, emergency braking, and sharing of road and traffic condition information.

The basic communication modes are distinguished as

- vehicle to/from roadside infrastructure
- vehicle to vehicle

Dedicated short-range communications (DSRC) have been set up to from the vehicle to infrastructure communications to support traveler information, commercial applications (toll/parking fee collection, in-vehicle advertising), and safety applications (intersection collision avoidance, approaching emergency vehicle warning, rollover warning) [51.14].

DSRC is a wireless protocol specifically designed for intelligent transportation systems. It is a subset of RFID technology, working in the 5.9 GHz band (US) or 5.8 GHz band (Japan and Europe). It is generally implemented with a dedicated protocol, uses short messages, and works in direct line of sight over short ranges (200–300 m).

For vehicle-to-infrastructure communications, a communication protocol architecture is currently being specified by the ISO [51.15], in the technical committee 204 working group 16 [51.16]. This protocol architecture is known as continuous air interface long and medium range (CALM) [51.17] and is being implemented in Europe under the cooperative vehicle infrastructure systems (CVIS) integrated project [51.18]. CALM is based on IPv6 protocols developed by the Internet engineering task force (IETF) [51.19], particularly the network mobility (NEMO) protocol, which allows sessions to be maintained between an in-vehicle Internet protocol (IP) network and the Internet backbone using any type of available media (3G, general packet radio service (GPRS), Wi-Fi, WiMax, M5, DSRC, satellite, etc.). For vehicle-to-vehicle applications, systems that support ad hoc networking are still in early stages. A vehicular ad hoc network (VANET) is a form of mobile ad hoc network (MANET) to provide communications among nearby vehicles and between vehicles and nearby fixed infrastructure equipment using direct and multi-hop intervehicular communication. Most of the research issues of concern to MANETs are of interest in VANETs, but the details differ. Rather than moving at random, vehicles tend to move in an organized fashion. The interactions with roadside equipment can likewise be characterized fairly accurately, and most vehicles are restricted in their range of motion, for example, by being constrained to follow a highway. The major work in this area is being supported by a series of workshops organized by Association of Computing Machinery (ACM) Special Interest Group on Management of Data (SIGMOD) [51.20].

Interaction between NEMO and the MANET or VANET is also under investigation but is not yet officially dealt with by any standardization body.

51.2.4 Digital Maps and Satellite Data

Combining GPS with stored digital maps creates a wide variety of intelligent vehicle applications [51.21]. Map data can greatly assist in the problem of road scene interpretation, map data can improve lane detection quality, help deal with the problems when sensors such as cameras do not work, e.g., in sunlight at dusk or dawn. Digital maps are widely used in commercial navigation systems for route guidance [51.22]. Such systems will be improved if real-time updating of the map occurred from traffic information [51.23]. Additional functionalities to be added include curve approach warnings [51.24], curve speed control [51.25], traffic sign information, and speed limit information [51.26].

51.3 Road Scene Understanding

Perception plays a key role in any robotic application. In the case of intelligent vehicles, the perception task is referred to as road scene understanding. It involves using different sensors (described in Sect. 51.2.1) combined with automatic reasoning, in order to create a synthetic representation of the environment around the vehicle. The knowledge base accumulated by this task is then used either to issue warnings to the driver in the case of advanced driving assistance systems (ADAS) or to control vehicle actuators in the case of complete autonomous driving. A complete and precise description of the state of surrounding environment is the key factor that allows the reduction of the number of false and missed alarms and provides the basis for smooth automatic driving. Needless to say, the perception of an outdoor environment – even if partially structured – is a challenging problem not only due to the intrinsic complexity of the driving environment itself, but also due to the impossibility of controlling many environmental parameters. Figure 51.5 shows examples of day/night, sun/streetlight illumination, temperature, poor visibility, rain/snow, and

different meteorological conditions, which in general are impossible to control and have to be faced by sensing devices.

The research community is addressing the issue of providing vehicles with robust and precise perception of the state of the environment from two different perspectives. One approach is to provide vehicles with ever-increasing sensing capabilities and processing power aimed at the provision of powerful onboard intelligent systems; Daimler-Chrysler is a world leader in this area [51.27]. An alternative approach is to use road infrastructure as an active component capable of communicating with all vehicles and sharing information on road conditions in real time [51.28]. Indeed these two perspectives can also be merged to provide a mixed solution to safely control a vehicle and in dynamic environments [51.28].

The task of road scene understanding may be addressed differently, depending on the availability of an intelligent infrastructure and on other players exhibiting cooperative behavior. The task of understanding the state of the environment can be simplified through the availability of information coming from other sources, thereby limiting the need to perform a robust and complete sensing onboard each vehicle. Helpful information could come from the infrastructure itself (for example, road conditions and geometry, number of lanes, visibility, road signs, or even real-time information such as traffic-light status or traffic conditions) or other players (such as the presence of the vehicle with precise position, speed, and direction). The players may also carry real-time information gathered by and shared with other players.

Although research is currently focussed on both intelligent vehicles and intelligent infrastructures, the first generation of production intelligent vehicles will have to rely primarily on their own sensing capabilities since the availability of information coming from other sources such as the infrastructure and other vehicles will take a while to be deployed in real-world situations. In fact, in order to be of practical use, intelligent roads must cover a large proportion of a country, and simultaneously cooperative intelligent vehicles must also be sufficiently widespread. It is important to note that the investment in intelligent infrastructures and intelligent vehicles comes from different sources: mainly from governmental institutions for the former, and vehicles owners for the latter.

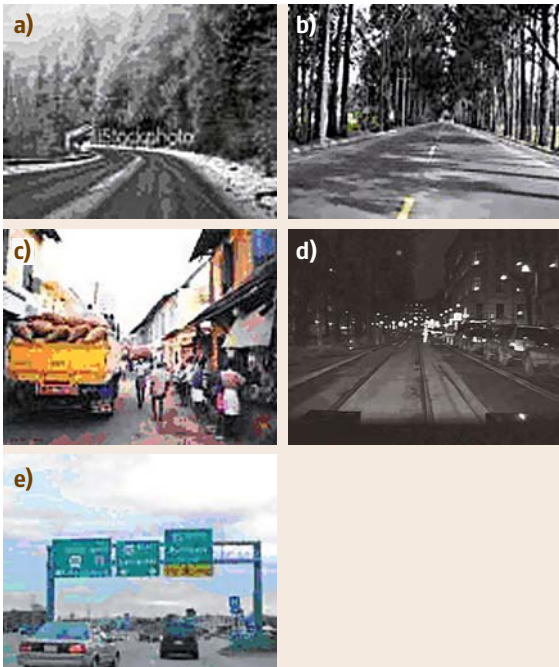


Fig. 51.5a–e Typical range of road scenes that intelligent vehicles must handle

The information that is owned by the infrastructure itself and that could be made available to the vehicles includes

- precise geometry of the lane/road
- road signs
- status of traffic lights

On the other hand, the infrastructure can also assess and deliver real-time data such as

- road conditions
- visibility
- traffic conditions

Another important piece of information that needs to be gathered by intelligent vehicles is the presence of other road players, such as

- vehicles
- vulnerable road users (pedestrians, motorcycles, bicycles)

Although it could be assumed that sometime in the future all vehicles will be equipped with active systems that allow them to be safely avoided by other vehicles, it is quite improbable that pedestrians and bicycles will have similar equipments: their presence will need to be detected using onboard sensors only. The same consideration also applies to obstacles that may unexpectedly be found on the road, or to temporary situations such as roadworks: if a vehicle needs to cope with the unexpected, then it needs to have the capability to assess the situation in real time with its own sensors.

This is why onboard sensing is of paramount importance for future transportation systems; vehicle-to-vehicle and vehicle-to-infrastructure communications may help and improve the sensing, but a complete sensor suite must also be installed on our future vehicles. The main challenges in road environment sensing are examined below.

51.3.1 Road/Lane Tracking

Many vehicles prototypes have been equipped with lane detection and tracking systems, starting from the very first implementations in the early 1980s [51.1–3]. Indeed in this case computer vision plays a basic role; although generally the road can also be detected with laser scanners [51.11], the only generic technology able to detect lane geometry and lane markings with high precision is computer vision. Most lane tracking approaches have focused on detecting lane markings and exploiting structure in the environment, such as the parallelism of

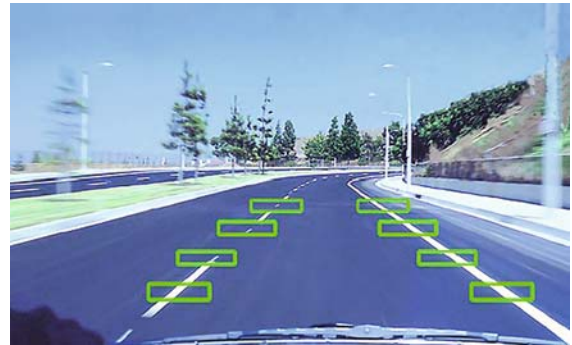


Fig. 51.6 Lane detection

the left and right lane markings, the invariance of road width, or the widely used flat-road assumption. These assumptions were mainly used to overcome the problem of having a single camera (a choice driven by cost). Some systems use stereo vision to detect lane markings and are able to work without such constraints. The problem of lane tracking in highway situations is basically a solved problem – with commercial systems being deployed in passenger and commercial vehicles [51.29]. An example of the typical output of a commercial lane tracker is shown in Fig. 51.6.

However, such systems cannot guarantee that lane detection systems will work with 100% reliability; these systems typically work with 95–99% reliability. Therefore, lane tracking systems are only being used in lane departure warning systems since no failures can be tolerated for autonomous driving. Efforts are underway to develop algorithms that will tolerate a variety of driving conditions, and push the 100% reliability boundary [51.30].

51.3.2 Road Sign Detection

Another fairly straightforward use of computer vision is road sign detection and understanding. Road signage is deliberately structured to aid human drivers. Road signs use a set of well-defined shapes, colors, and patterns. The signs are placed at consistent heights and positions in relation to the road. Therefore reading road signs is an achievable task for computer vision. Detection is done using a collection of shape and/or color detection schemes [51.31,32]. After the detection and localization phase, recognition takes place. Normally this task is performed by pattern-matching techniques such as image cross-correlation, neural networks, or support vector machines since the possible set of road signs is limited and well defined. Figure 51.7 illustrates the con-

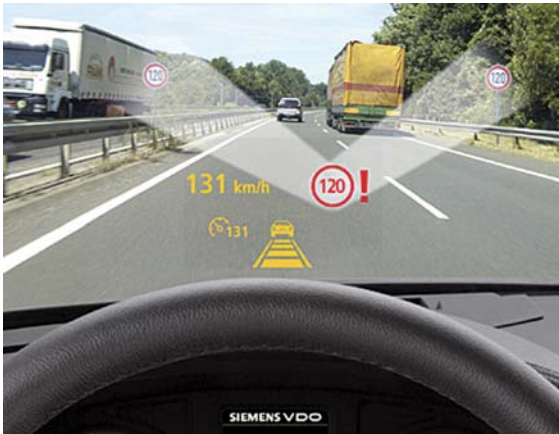


Fig. 51.7 Road sign detection for speed warning application

cept of a speeding warning system based on speed sign detection. The challenge for research work in this area lies in the robustness of detection and the reliability of classification of signs. Most automotive companies have systems under development, e.g., Siemens [51.33].

51.3.3 Traffic-Light Detection

Color and pattern matching are also the key techniques used for the detection of traffic lights [51.34]. Although the detection of traffic lights is not overly complex, this application hides one further aspect that makes vehicular applications difficult: besides the correct localization and recognition of a signal, special care has to be taken in checking the signal position and orientation on the road/lane since that signal may not be addressed to the current vehicle. This is particularly true in downtown intersections at which many traffic lights are visible at the same time; in this case the vehicle must have the capability to select the correct traffic signal that must be obeyed. Some experiments have been undertaken with active traffic signals, able to emit the status of the traffic light using radiofrequencies [51.35]. This involves additional infrastructure; at this stage vision seems to be the only simple viable solution.

51.3.4 Visibility Assessment

One of the key challenges is detection of fog. The meteorological visibility distance is defined by the International Commission on Illumination (CIE) as the distance beyond which a black object of an appropriate size is perceived with a contrast of less than 5%. Different techniques for measuring this parameter –

and thus detect foggy conditions – have been implemented [51.36]. Although many of the methods use vision, there are also efficient alternatives – generally used in fixed locations such as airports and traffic-monitoring stations – based on the use of multiple scattering lidars. The main challenge of using vision to estimate visibility is that a moving vehicle generally cannot rely on a specific reference point/object/signal at a specific distance.

51.3.5 Vehicle Detection

The detection of vehicles has been addressed using a large variety of sensor technologies, ranging from vision to lidars, from radars to sonars.

Despite being different in shape and color, vehicles share the same characteristics and feature a large size and reflective material. The position of vehicles is predictable once a rough indication of the road/lane position is available. Vehicles, in fact, can be successfully detected by many different sensors independently [51.37–39]. Figure 51.8 shows a vision-based vehicle detection system.

Nevertheless, although the solution to this problem seems straightforward, each sensor has its own domain of application and its own challenges. Vision is generally powerful, but may fail in low visibility and bad-illumination scenarios (night or tunnels) or in heavy traffic conditions when vehicles may occlude each other. Vision in the infrared domain (thermal imaging) is able to detect vehicles with a high confidence since vehicle tires and mufflers generally exhibit high temperatures and are therefore easily detected in the image. However, parked vehicles, trailers, and even vehicles which have just started to move are colder than running vehicles and therefore less visible. Lidars are generally robust, but have decreased sensitivity in adverse weather condi-

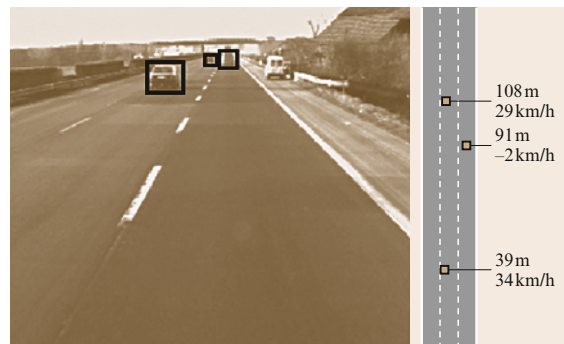


Fig. 51.8 Vehicle detection – lane and range position

tions. Radars, while cheap, can suffer from bias in lateral measurements due to the presence of other nearby reflecting objects. Finally sonars are applicable only for very short distances. The research challenge is to implement multisensor fusion robustly. A common approach is to fuse vision with radar [51.40].

51.3.6 Pedestrian Detection

The detection of vulnerable road users (pedestrians and bicycles) is one of the most difficult tasks for intelligent vehicles. The appearance of a pedestrian is challenging: a pedestrian shape can change greatly within a few tens of milliseconds, there are no clear invariants in color, texture, or size, and no assumptions can be made about posture, speed, or the visibility of parts of the human body such as the head. Machine learning methods have been successfully applied to this problem [51.11, 27, 31, 41]. Greater reliability and reduction

in false alarms have been achieved through the incorporation of stereo vision [51.42]. However, the detection of vulnerable road users is one of the most relevant research topics worldwide since a great number of benefits – including insurance reductions – may be achieved once a fully functional pedestrian detector is available on cars. Countermeasures may be activated to reduce the consequences of vehicle–pedestrian accidents, such as the firing of external airbags or the opening of the hood to lessen the impact of a head-on collision. Currently, with all possible technologies under evaluation, no solution seems to offer reliable detection in every scenario: radars are not able to detect pedestrians reliably in crowded scenarios, while vision has the many drawbacks listed above. Even thermal imaging, which – although still very expensive – is generally believed to be one of the most promising technologies, fails in some situations, such as hot summer days and, in general, in high-temperature environments.

51.4 Advanced Driver Assistance

Given the legal liabilities and technical challenges of achieving 100% reliability for autonomous intelligent vehicles, it appears likely that motor vehicles will have pieces of autonomous functionality added progressively and that cars will eventually evolve into autonomous robots. The perception techniques described earlier can be used in a variety of ways to make driving safer, more efficient, and less demanding. Individual perception techniques, or combinations of sensing modes, are being used to provide warnings to drivers of dangerous situations. These warnings are being used to prevent collisions in a variety of situations, such as when backing up, when leaving a roadway, into rear ends, on lane changing/merging, with pedestrians, and at intersections.

Developing a collision warning system requires many steps beyond building the perception system. For a typical example, roadway departure warning, the steps followed in a program initiated by the US National Highway Traffic Safety Administration include:

1. Statistical studies: In the United States, crashes involving a single vehicle leaving the roadway are relatively rare, but disproportionately dangerous; approximately 40% of the 40 000 fatal crashes per year in the US are single-vehicle crashes where the vehicle leaves the roadway. The first part of the study
2. Causal factors: The second step was to determine the causes of those crashes. Most run-off-road crashes are due to driver factors such as excessive speed, inattention, or loss of control. This is an important observation, because it means that alerting the driver, or warning of difficult situations, could prevent those incidents. For the fraction of crashes caused by mechanical failure a warning system would not be useful; in this type of crash, mechanical failure is involved in less than 5% of the crashes.
3. Opportunities for intervention: This part of the study set out to determine whether a warning system could be effective, and, if so, how far in advance the warning would have to be given. Given typical road departure trajectories, typical widths of roads and shoulders, and the range of potential steering responses, this task generated requirements for how accurately the system would have to track vehicle trajectory in order to predict a roadway departure.
4. Human factors: Since the system being designed is a warning system, rather than an active control, it is crucial to understand what kinds of warning a driver (who may be distracted or sleepy) would respond to, and how quickly and accurately the response will

be. Reaction times vary widely across individuals: using one second for reaction time is a fairly standard estimate.

5. Simulator studies: A driving simulator is like an aircraft flight simulator for cars or trucks, with a variety of simulated roads and conditions. Simulator studies were used to test driver response to warnings: directional or nondirectional audio warnings, steering wheel shakes, and combinations. A nondirectional audio cue worked best.
6. System specification development: Based on the preceding steps, in order for a system to be useful, it needs to work at day and night, in almost all weather conditions, needs to measure vehicle speed, lateral position relative to the road, lateral heading, and roadway curvature, and predict future vehicle trajectories long enough in advance to trigger a warning alarm.
7. Perception and system development: Given those specifications, there are several ways that a perception system could be built to sense the road and the vehicle's trajectory relative to the road. For this particular test, a lane keeping system – rapidly adapting lane position handler (RALPH) was developed and tuned [51.43].
8. Limited tests: The entire system – sensors, processing, driver interface – was built and tested with a limited number of volunteers, and the system tuned and validated.
9. Full-scale operational tests: The system was deployed in test fleets, including long-haul trucks and passenger cars.

51.4.1 Collision Avoidance and Mitigation

The complete cycle from the idea of using perception to prevent crashes, to full system development, took over 10 years in this case. The pure *robotics* part of the system is a crucial element, but is only one piece of the development needed to make a useful product. Some active control has already been assumed by today's vehicles. Antilock brakes have been on the market for many years. Traction control systems which control throttle to stop wheel spin are being introduced. Electronic stability control systems take this the next step further, controlling throttle and individual wheel brakes to help in cornering performance. So, gradually, people are willing to cede some control to very reliable automated systems. We can expect this trend to continue.

Each collision warning type has its own list of specific development challenges, as described below.

Backup Collisions

The sensing challenge is to see relatively small objects, such as fence posts or children's toys, while not picking up false alarms from pavement joints or leaves and debris. The sensors used in today's commercial vehicles are piezoelectric ultrasonic sensors, which are inexpensive [51.9]. However, ultrasonic sensors have well-known limitations. The challenge of developing low-cost, accurate, reliable sensors remains.

Rear-End Collisions

These are among the most difficult collisions to prevent, with the most challenging sensing conditions. Rear-end collisions often happen at high speeds, requiring long-distance sensing of other vehicles (up to 100 m at US highway speeds, much longer at the high speeds found on some European roads or for the longer braking distances needed for heavy trucks). That in itself is not too demanding a challenge: the sensed objects in this case are relatively large and have high metal components, so radar and lidar are both feasible sensing modes. The biggest range-sensing challenge is sorting out true targets (slow or stopped vehicles) from false targets (overhead signs or bridges, and side lobes from strong reflectors on the side of the road). It is also important to determine if the sensed vehicle is in the same lane as the smart car, or a different lane. Sensing lane markings at such a large distance is a very difficult challenge; merging lane sensing (often done by vision) with obstacle sensing (by a different sensor) and registering the two to within the resolution of a lane is a daunting task. This technology remains under development through industry and government programs [51.44].

Lane-Change/Merge Collisions

In the simplest case, the countermeasure to this kind of collision involves short-range sensing to cover the *blind spots* on the rear corners of a vehicle, where it is difficult to see with mirrors. For passenger cars, this area is quite small, and can be covered with a single sonar or radar. Often, the user interface is a warning light placed in the rear-view mirrors [51.45]; this reinforces good driver behavior of checking mirrors before changing lanes. The sensing challenge for heavy trucks or transit buses is the same as for cars, except that the area not visible in planar mirrors can be much larger. Figure 51.9 shows examples of blind-spot detection for cars and heavy vehicles.

The usual solution is a row of sensors along the side of the vehicle, although scanning lidars or panoramic vision are also used in some experimental applica-

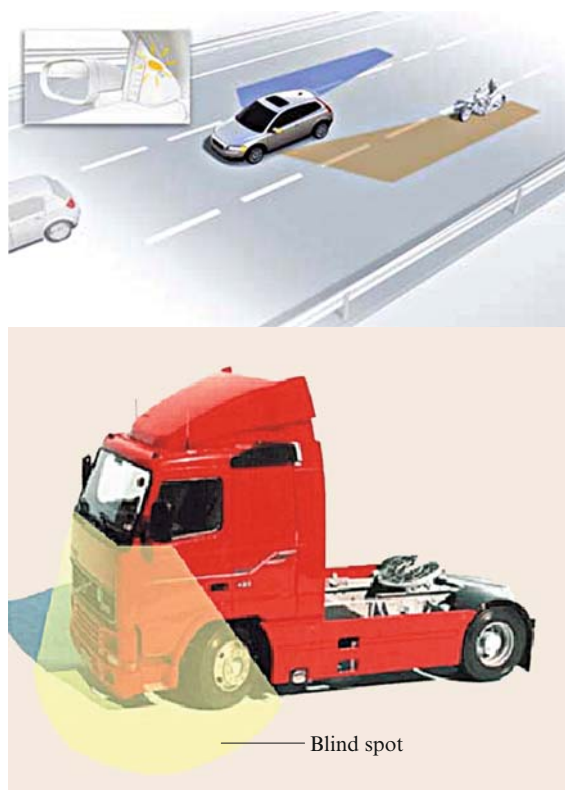


Fig. 51.9 Blind-spot detection

tions [51.46]. The further complication for lane-change warnings is in high-speed driving, where it is important to look not just adjacent to the vehicle but a long way to the rear, to find overtaking vehicles with high relative speeds. Recently a commercial product has been brought to market [51.47].

Pedestrian Collisions

Pedestrians are particularly important to detect, because pedestrians are much more vulnerable than people in vehicles; as discussed earlier they are also unfortunately relatively difficult to detect and very hard to predict. Just detecting a pedestrian is not sufficient. In transit operations, for instance, a bus operates close to pedestrians much of the time. To do meaningful collision warning, it is important to detect the pedestrian, detect their current path, look for cues such as crosswalks or curb edges that modify the probability of the pedestrian's trajectory, and match all of these factors with the predicted trajectory of the vehicle. It is crucial to tune the warning system to produce few false alarms while not missing real alarms. A particularly dangerous situation is pedes-

trians slipping and falling underneath a bus: these are very dangerous situations, but very difficult to detect in time to warn the driver. For these reasons, automotive manufacturers have worked on products such as night vision to enhance driver perception [51.48].

Intersection Collisions

Intersection collisions are particularly difficult to prevent because they often involve challenging sensing scenarios. Many of these collisions involve occluded vision, with lines of sight blocked either by large vehicles or by adjacent buildings. They also often involve high closing rates from oblique angles, making it necessary to see a long distance with a very wide field of view. The solution usually proposed is to add intelligence to the infrastructure, either in fixed sensing (such as radars looking down each approaching road) or in some kind of radio relay that takes data from approaching smart cars and passes it to other approaching vehicles. None of these solutions is particularly attractive: the large number of intersections makes it difficult to envision any universal solution.

Other Obstacle Collisions

Vehicles have collisions with many things other than other vehicles and pedestrians: animals (deer, dogs, cats), car parts (tire carcasses, rusted-out exhaust systems), cargo that falls off of trucks, construction debris, etc. Warning drivers about these kinds of objects on the roadway is a challenging task. A piece of construction timber on the roadway may be large enough to do significant damage to a car, but be small enough to be difficult to see, and be invisible to radar. Some interesting work has been done with high-resolution stereo vision [51.49], with polarimetric radar [51.50], and with high-resolution scanning laser range-finders [51.51]. However, in general this remains a difficult problem.

Other Actions

Besides warning the driver, there are other actions that an intelligent vehicle can take short of assuming control. If a collision is inevitable, particularly from the side of the vehicle where there is limited crush space, the system can brake and deploy airbags even before physical contact. Of course, such a system would have to be nearly 100% reliable. More simply, if the system senses an imminent front collision, it can preload the power brakes, saving fractions of a second in brake reaction time. The driver must still actuate the brakes, but the onset of hard braking can be much quicker. At 100 km/h, a 0.1 s saving in braking actuation saves approximately 3 m of

stopping distance, which can be the difference between a severe rear-end collision and a much lighter crash. Such systems are being introduced into the high-end market by all the major automotive manufacturers.

Collision Avoidance

The next step beyond emergency braking is an automated system. Such systems have several advantages over a human driver: much quicker reaction times, access to sensors such as individual wheel speeds and slips plus external sensors such as radars or lidars, access to individual brake controls and other controls, and so forth. So, if the system had ideal situational awareness, it could in many cases do a better job of avoiding a collision than a human could. This is still a very difficult area for implementation, however. First of all, the human has access to higher-level knowledge: the driver may be watching the behaviors of other cars, may make eye contact with pedestrians or drivers, may be watching a policeman directing traffic, etc. So it may be that a manoeuvre that, to the system, looks like the best way to prevent the collision, is actually the wrong action to take. Secondly, in most countries, as soon as the vehicle takes control there is a shift in the liability for any resulting crash from the driver to the manufacturer. So there is a great reluctance to take active control. An alternative approach recently developed is to observe the environment state with lidar, and monitor the dynamic vehicle state to determine whether the accident is unavoidable. If the driver can no longer take corrective action, that is, brake or turn away safely, then emergency braking occurs [51.11].

For now, however, active collision avoidance remains a research area, with significant questions of reliability, human factors, and liability.

Combining perception with control gives partial automation for specific tasks, such as adaptive cruise control, lane keeping, assisted parking, and slow driving in stop-and-go situations.

51.4.2 Adaptive Cruise Control

Adaptive cruise control (ACC) is the logical extension of standard cruise control to also include keeping a safe distance from the preceding vehicle. If there are no vehicles in front of the smart car, it follows a set speed, as with standard cruise control. If a slower-moving vehicle is in front, an ACC-equipped car will sense the vehicle using radar or lidar, and slow to maintain a safe distance (typically set to a 1.5–2 s following gap). Figure 51.10 shows an illustration of the ACC concept. The sensing

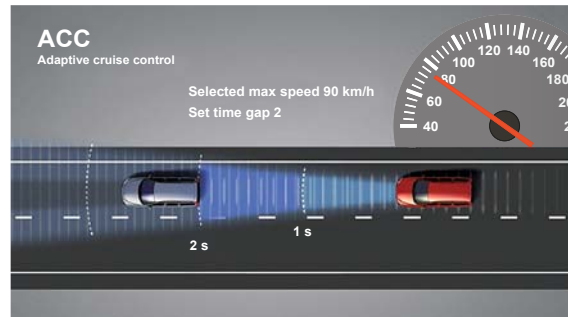


Fig. 51.10 Advanced cruise control (ACC)

challenge for ACC is much easier than the challenge of rear-end collision countermeasures, since ACC systems are only designed to deal with other moving vehicles.

The biggest sensing difficulty for rear-end collision countermeasures is separating stopped vehicles on the road from objects off the road; for ACC this difficulty is bypassed by ignoring all stopped objects. Moving objects are classified as in-lane or out-of-lane based on a number of heuristics. Often, the smart car's own steering radius is used as an estimate of the road curvature ahead, in order to determine whether vehicles ahead are in the same lane. Since the systems are explicitly sold as *convenience* instead of *safety* systems, they only need to deal with normal situations with relatively low differences in velocity, and they leave the more difficult situations up to the human driver. The human is still alert, controlling the steering, and watching the traffic. These systems are being introduced by all the major car manufacturers.

51.4.3 Stop and Go

Stop-and-Go driving assistance (also referred to as low-speed ACC) is at the opposite end of the speed spectrum, when vehicles are creeping along in dense traffic. At slow speeds, it is easy to track the vehicle ahead, and to move when it moves, steer when it steers, and stop when it stops. If the traffic accelerates to a modest speed, the stop-and-go system disengages, and the human must assume control of the throttle, brake, and steering. Since the speeds are very slow and the distances are short, many different sensing systems will work such as stereo vision, radar, and lidar [51.52, 53].

51.4.4 Parking Assist

Parking assistance is also a low-speed aid. In a typical scenario, the driver initiates the system by pushing

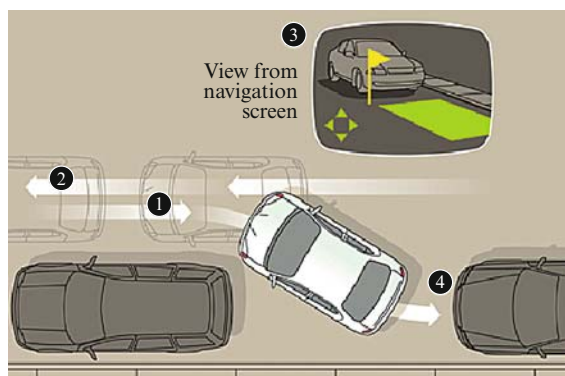


Fig. 51.11 Parking assistance

a button when driving past an empty parking space. The system measures the length of the space using odometry, measures the positions of the cars in front and in back using short-range sensors, and infers the position of the curb by assuming that the surrounding cars are standard-sized cars parked near the curb [51.54]. Figure 51.11 illustrates parking assistance.

Once the system is fully engaged, it takes over steering, planning and executing the ideal parallel park steering sequence. In some systems, the human is still responsible for throttle and brake, again insuring that the human is alert, watching for encroaching pedestrians or other obstacles. Such a system has been introduced by Toyota [51.55].

51.4.5 Lane Keeping

Lane keeping assistance is the natural extension to road departure warning systems. Given a lane tracking system, it is straightforward to add control of the steering wheel to keep the vehicle centered in its lane [51.1–3]. This has a number of uses. Some cities would like to run transit buses on narrow roadways, for instance the shoulder of highways or through narrow streets in old cities. Automated lane keeping systems using mechanical guideways are in use in several places [51.56]. It is easier and less expensive if such systems can be electronic rather than relying on specially installed mechanical guides. A specific subcategory is precision docking: for a transit bus to pick up a passenger in

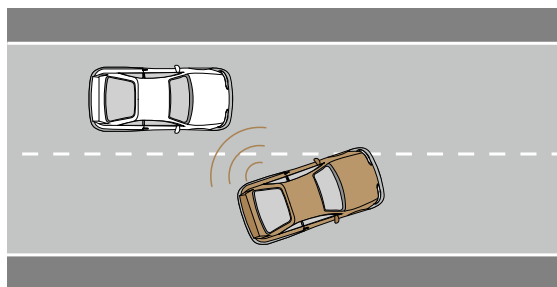


Fig. 51.12 Lane changing – side detection

a wheel chair, either the bus must deploy a special ramp (which is a slow process), or it must pull up to a level dock and leave a very small gap, so that the wheel chair can safely roll on or off. Short-range precision docking systems use either a downward-looking sensor, looking at painted lines or magnetic markers, or a sideways sensor looking at the curb or dock, in order to guide the bus to its parking spot. Finally lane-keeping assistance is a convenience for driving on highways, especially with gusty winds. Honda has released a vehicle equipped with both lane-keeping assistance and ACC [51.57]. The danger with such systems is that the driver no longer has an active moment-to-moment role, and may lose concentration or even fall asleep. These systems are not designed for full automation, and still require a driver to handle unusual circumstances. The next stage is to integrate driver state monitoring. If the driver is inattentive then all automatic systems are disengaged.

51.4.6 Lane Changing

Lane changing assistance is the next extension in partial autonomy. It combines lane keeping and ACC with blind-spot monitoring. If the car can safely overtake a vehicle, then a lane change is undertaken and speed is unchanged. Otherwise, the ACC slows the vehicle down. At its simplest such a driver assistance system can advise a driver whether a lane change can be safely undertaken [51.58]. In its most advanced form the lane change is undertaken completely automatically by the vehicle [51.1]. Such systems require an additional side-facing sensor, typically radar (shown in Fig. 51.12)

51.5 Driver Monitoring

There has been an evolution of thought regarding the role of the driver in intelligent vehicles. The grand goal

has been to replace the driver with a fully automated system. As discussed earlier, full automation of intel-

Intelligent vehicles is still some years away due to system reliability and legal liability reasons. The next step in the development of the motor vehicles is partial automation – where individual autonomous functions such as ACC, lane keeping, lane changing etc. are developed. Motor vehicle designers have realized that the driver cannot be removed from the vehicle, and must instead be supported by systems onboard a motor vehicle. Over 92% of motor vehicle accidents are caused by driver error [51.59]. It is likely that the next generation of intelligent vehicles will work in the following way.

1. The vehicle will monitor the road scene using advanced driver assistance system technologies discussed earlier to assess the state of the environment and warn the driver of dangerous situations, e.g., lane departure warnings.
2. The vehicle will also monitor the driver using vision sensors to assess the state of the driver. If a driver is fatigued, drowsy, inattentive, distracted or under the influence of drugs then accidents can occur. The vehicle warns the driver of dangerous circumstances, e.g., drowsiness warning.
3. For legal liability reasons intelligent vehicles will not take control, rather the driver will be alerted using visual, audio, or tactile warnings. The vehicle will not perform collision avoidance; rather collision mitigation will occur through emergency braking.
4. If an accident is unavoidable the vehicle can autonomously apply emergency braking. To maximize the safety of the occupants in the vehicle, seat-belt restraints are tightened and airbags are safely deployed.
5. After an accident has occurred knowing the state of the driver and the passengers is important. If an occupant has been injured a call to the emergency services can be dispatched automatically by the vehicle.

In all the steps described above monitoring the driver is critical. For future advanced driver assistance systems (ADAS) to work safely the driver should be put in the loop, for example, in a lane departure warning systems, it is not possible to determine whether a vehicle departing from a lane due to cause by driver intention or error. If the state of the driver is being monitored, and the system can detect that the driver's eyes are closed or the driver is looking away from the road, then it can be inferred that the lane departure was involuntary and that a lane departure warning should be given to the driver. For ADAS to be accepted by drivers the systems should not give false warnings. If the driver is looking directly at the road

then lane departure warnings should not be given (or a different subtle warning should be given). Similarly more sophisticated systems such as lane keeping should not be engaged unless the driver is fully attentive and has their hands firmly on the steering wheel. The key point is that drivers must be fully engaged with and in control of the driving task. This is a most important consideration in the design of ADAS for intelligent vehicles.

Combining perception with control gives partial automation for specific tasks, such as adaptive cruise control, lane keeping, assisted parking, and slow driving in stop-and-go situations.

51.5.1 Driver Fatigue, Inattention, and Impairment

By directly monitoring the driver using visual sensing opens up the possibility of developing a new class of ADAS applications. It is possible to monitor driver state through monitoring signals such as an electrocardiogram (ECG), temperature etc. However, market studies by automobile manufacturers have shown that people do not like any wires, or gadgets attached; driver monitoring must be noncontact and noninvasive. The only solution is to use vision as the sensing medium. The technical challenge to develop a computer vision system that can automatically detect a driver of any age, sex, race, with/without eye or sunglasses, and with/without facial hair is enormous. Recently significant progress has been made with systems being developed that can also detect where a person is looking (gaze direction) [51.13].

Once the driver's state (head position, eye gaze, eye blink rate) can be measured then ADAS applications



Fig. 51.13 Driver state detection

can be developed. Figure 51.13 shows the output of a commercial driver state detection system.

Driver Impairment

Safety authorities estimate that as many as 50% of all road accident fatalities are caused by driver impairment due to alcohol or drugs [51.60]. Recent research has shown that driver impairment can be detected by sensing abnormal scanning patterns of eye gaze. It promises to open up a new class of ADAS. There have been major education and legislative initiatives in many Organization for Economic Cooperation and Development (OECD) countries, resulting in a significant reduction in road fatalities. However, the difficult cases (fatigue, distraction, and inattention) have become more prominent. ADAS technologies could have a significant impact in further reducing the road toll.

Driver Fatigue

Safety authorities estimate that 25–30% of all road fatalities are caused by driver fatigue [51.61,62]. Research has shown that there are four visible factors that indicate the onset of fatigue – prolonged eye closure, uncontrolled head moments, drooping eyelids, and reduced eye-gaze scanning. Systems are under development that focus only on eye closure [51.63]; the challenge of fusing all four factors together into a robust algorithm that works for a wide range of drivers remains an open research problem.

Driver Inattention

Safety authorities estimate that up to 45% of all traffic accidents – from minor dents to serious incidents – are caused by driver inattention or distraction [51.64]. Research has shown that if the drivers consistently keep their eyes on the road then driving becomes a much safer experience [51.65]. All major automotive manufacturers are developing ADAS applications that warn the driver if they are distracted from the driving task. More sophisticated ADAS under development use the driver's gaze direction to check whether safe driving practices are being followed, for example, did the driver check the side mirror before a lane change was performed? If the driver fails to check the side mirror a warning would be issued.

Driver Workload

The increase in new electronic systems and gadgets that are being installed into today's motor vehicles

is also another source of distraction. Questions arise about when and under circumstances should a driver change a compact disc (CD) or answer a phone call. Research is underway into the development of workload systems. These systems take into account the vehicle state (speed, acceleration, braking, gear-change yaw rate etc.) to determine whether information management tasks such as answering a phone call, sending a short message service (SMS) message are allowed [51.65]. The next stage of research is to include state information about the road scene and the driver. If the car is driving on a country road, and the driver is attentive then distractive tasks could be allowed and managed.

51.5.2 Driver and Passenger Protection

The automotive industry is moving towards the use of active safety systems such as airbags to complement passive systems such as seatbelts. ADAS will play a critical role in active safety systems. In emergency braking or impending rear-end collision situations, seatbelts can be pretensioned and airbags primed. Drivers and passengers could be further protected through the development of smart airbags [51.66]. Airbags, while considered to be an essential part of a modern car, can cause fatalities if the occupant is too close to the airbag, is a child, or is not wearing a seatbelt. Smart airbags deploy that are dependent on the position of the vehicle occupant's head. The driver state monitoring technologies discussed earlier are an essential component of smart airbags.

51.5.3 Emergency Assistance

The speed at which people are given treatment after a serious accident is critically related to the survival rate of victims. While it is expected that ADAS will lead to a reduction in road accidents, it will not eliminate accidents. Therefore it is important that automated systems for emergency situations are developed. Systems that use GPS, vehicle state information, and mobile communication systems have been developed that send the world coordinates of the vehicle to emergency authorities after an accident [51.67]. This information can be augmented by using driver monitoring technology to assess the condition of the occupants of the vehicle. Such types of system are currently being researched.

51.6 Automated Vehicles

Intelligent vehicles that are fully autonomous can be justified for safety, traffic congestion, and environmental considerations.

51.6.1 Operating Safely

The first problem is safety. As stated earlier, automobile accidents are one of the main causes of human fatalities – on a staggering scale. This is a catastrophe of larger magnitude than all armed conflicts since the beginning of mankind. The most economically advanced countries (the OECD) have been able to reduce the number of fatalities significantly through improved technology in vehicles – improved handling, braking and passive safety. Transport infrastructure has been greatly improved; modern freeways are ten times safer than regular roads [51.68]. However, these improvements appear to be reaching a limit in terms of the number of deaths per million passengers-kilometers, particularly in industrialized countries.

As discussed earlier, motor vehicles are inherently dangerous due to their reliance on human control. Slight errors at high speeds can have catastrophic results. As discussed earlier the most common cause is driver distraction, which leads to improper reaction times or driving actions. Human error also frequently occurs while handling emergency situations. A large percentage of drivers will take improper action in such situations and produce an accident that could have been avoided by a skilled and attentive driver [51.68]. The best solution to these problems is to remove the driver from the control loop. As discussed earlier the interim step is to assist the driver to warn him/her in case of potential danger (e.g., in the case of excessive speed before a dangerous bend or when a car is present in the blind spot while changing lane), or to take over control in emergency situations (e.g., emergency braking in the case of impending collision). For legal liability reasons, until it can be shown that autonomous systems have high integrity and reliability, people must be kept in the loop – in a supervisory capacity.

51.6.2 Traffic Congestion

The success of the automobile also leads to the saturation of the road infrastructure, particular in cities. Each car needs a certain amount of space in order to operate safely. The usual width of roads is 3.5 m in order to accommodate steering imprecision, while vehicles

are about 2 m in width. Spacing between vehicles also has to be kept at a safe minimum to prevent collisions during deceleration (this principally depends on driver reaction time). It is usually recommended that the spacing should correspond to a time gap of at least 1.5 s. This spacing leads to a maximum throughput of about 2200 cars per hour per lane, independent of traffic speed. This is not high if we consider that a suburban train can carry about 60 000 passengers per hour on an infrastructure of similar dimensions. Furthermore, high-density car traffic of greater than 2200 cars leads to a breakdown in traffic flow (stop-and-go traffic) and to increased likelihood of accidents, which can drastically decrease the overall capacity of system. The solution to this problem also lies in the removal of the driver from the control loop to improve lateral guidance (reduction of the width of lanes) and longitudinal control (with possible time gaps of around 0.3 s, independent of the speed) while maintaining traffic safety. Such techniques of automated driving could multiply the throughput of road infrastructure by a factor of five. This was demonstrated in particular by the work performed in the advanced highway systems (AHS) project in the United States, which led to the demonstration of seven-car platoons running at speeds up to 130 km/h on a dedicated freeway in San Diego with gaps of about 0.5 s [51.69].

Another congestion problem is associated with parking. Every individual vehicle is only used for a small percentage of its total usable life. Most of the time, motor vehicles occupy space very unproductively. Typically, a car requires about 10 m² of space. Usually, parking occurs at the curbside – a space that is very limited in large cities and cannot accommodate all the cars of residents and visitors. In parking lots, each car will need four times this amount in order to have access to each individual slot. If a transportation system based on fully automated cars could be developed, people would not need to own cars but could rely on a service such as the one offered by taxis with vehicles that would come on demand and offer a complement to mass transport. This is the concept of the cybercar, which is under development in Europe [51.70].

51.6.3 Environmental Factors

The ever-increasing deployment of passenger and commercial vehicles has led to critical environment problems of noise and of pollution in the local community. In addition greenhouse-gas emissions have an impact at the

global environment level. Recently automotive manufacturers have been able to reduce emissions of local pollutants drastically; noise in cities is now perceived as the major problem by the inhabitants. At the global level, the generation of CO₂ through the use of fossil fuels is also considered to be a major problem that will require drastic steps. In the short term, this may lead to limiting the use of vehicles that generate CO₂ above certain levels. In the longer term, this will lead the automotive industry to offer vehicles that run on various forms of energy and to new forms of transportation systems with much higher efficiency. Automated vehicles running in platoon formations on new infrastructures could form such a system.

51.6.4 The Automobile of the Future

All three challenges are now at the heart of new policies being developed in many countries. These policies concern safety and emissions features on the vehicles with a strong push towards advanced safety systems, as was recently seen in Europe with the *Intelligent Car Initiative* [51.71]. At the infrastructure level, there is also a strong push to implement regulation schemes to limit and control the use of road transport and promote alternative and more efficient transport means. In the future, this means that the use of a private automobile will be much more regulated and more integrated with other modes of transport.

The automotive industry might move from an industry of products to an industry of service where anyone can have access to mobility in the most cost efficient way. Both mass transportation and individual transportation would be offered by companies in the most cost-efficient way, respecting local regulations. Companies operating in this service industry could be transit operators, taxi



Fig. 51.14 Car sharing from Toyota



Fig. 51.15 Automated electric cars from Honda

companies, car rental companies, car-sharing companies or even new entrants into the transportation business such as mobile phone companies, which are already familiar with large customer bases and mobility services. Car-sharing operators such as those operating in Germany, Switzerland, Japan, and the USA are likely candidates [51.72]. Figure 51.14 shows a commercial car-sharing operation.

In this context, new types of vehicles such as electric ones and automated driving are being developed because of the decrease in cost of operation and improvements in safety.

Demonstration projects by Honda and by Toyota have already been put in place along these lines. However, this market is still searching for its operators and business model as well as the right products [51.70]. Figure 51.15 shows the Honda concept automated car.

There are two distinct trends in the future of automated vehicles. One is with advanced driving assistance, which has spread rapidly since the late 1990s with numerous techniques appearing recently in high-end passenger vehicles and commercial vehicles (buses and trucks). This trend has been described earlier in this chapter. The other trend is associated with the arrival of people-movers based on automated guided vehicles (small or large) in specific locations and on dedicated tracks (protected or not).

It is forecast that, in the next ten years, these two trends will merge, with individual vehicles having dual-mode capabilities: manual (with strong control and assistance) driving on regular roads and fully automatic driving on dedicated zones where no (or few) manual vehicles will be allowed, thereby ensuring smooth and safe operation of the automated vehicles [51.70]. This type of vehicle will be perfect for the implementation of mobility services with vehicles that can be called on

demand (perhaps through a mobile phone) when and where needed. With the development of such zones, new dedicated infrastructures will be built specifically for these vehicles to go automatically and at high speed from one automated zone to the next. This appears to be the most realistic path for *automated highways* to be realized, since it is now considered nearly impossible to have a smooth evolution from the actual infrastructure with its manual vehicles to one with a majority of fully automated ones. This is one of the reasons why the AHS project was cancelled in the USA despite a very successful demonstration in 1997.

51.6.5 Automated Vehicle Deployment

As discussed earlier some automated functions are being introduced in production vehicles. Honda and Toyota have introduced a combination of lateral control (lane keeping assistance using image processing) and longitudinal control (adaptive cruise control using laser and radar sensors) [51.57]. An intelligent parking assist sys-



Fig. 51.16 Toyota ABRT used in IMTS Phileas



Fig. 51.17 Cybercars at Schiphol airport, Amsterdam

tem has been introduced recently by Toyota, offering the ability for the car to be parked without the driver using the steering wheel. However, this system does not use sensors: the driver has to position his/her vehicle on the image of the rear camera [51.55]. However, most car manufacturers and some component manufacturers are actively working on the sensors to remove this task from the driver.

Fully automated vehicles without any human intervention or supervision are now appearing in the commercial domain. Automated bus rapid transit (ABRT) combines the service quality of rail transit with the flexibility and cost of buses (as shown in Fig. 51.16). Nonautomated bus rapid transit (BRT) is already recommended by the World Bank in developing countries as the most efficient mass transport system. By reserving a dedicated lane for bus operation and adding some light infrastructure to facilitate loading and unloading, capacities similar to those of a train (60 000 passengers/h/direction) can be obtained, as has been demonstrated in South America [51.73]. By adding

the driving automation on a [BRT](#), the system can be made more efficient and safer, as is already the case with automated metros.

The intelligent multimode transit system (IMTS) consists of vehicles guided by magnetic markers imbedded in the middle of their dedicated roads. The platoon running function (three electronically linked vehicles running in file formation at uniform speeds) of the IMTS consists of precisely controlling the speed of all the vehicles in the platoon to be the same at all times [51.74]. In the city of Eindhoven in The Netherlands, multi-articulated buses with several steering axes are also running in automated mode on a dedicated track using magnetic markers [51.75]. These [ABRT](#) usually keep a driver on the vehicle since there is the possibility of reverting to the manual mode if need be (such as for an unexpected obstacle or in situations with many pedestrians); however, future plans are for full automation.

[ABRT](#) technology can also be found with smaller vehicles, now called cybercars, for on-demand door-to-door operation [51.70]. These vehicles have been put in operation for the first time at the Schiphol airport (Amsterdam) in December 1997 to move passengers from long-term parking lots to the airport terminal (Fig. 51.17). Since this time these cybercars have been

further developed, financed by the Information Society Technologies (IST) programme of the European Commission [51.70]. The long-term future of cybercars may lie with the development of dual-mode vehicles, with a particular emphasis on car-sharing operations [51.72]. Cybercars will have an automatic mode for operation in city centers (restricted to this type of vehicles) and a driver-operated (with assistance) mode for regular roads. The automobile industry is examining the viability of the development of such vehicles.

In order to match the performance of manually driven vehicles in urban environments, automated vehicles face major technical challenges. In particular, planning a trajectory for a car while avoiding moving and static obstacles remains as an important research challenge. In 2005, the [DARPA](#) Grand Challenge described at the beginning of this chapter brought together a large number of these researchers to demonstrate the feasibility of automated driving techniques. Five vehicles succeeded in completing the difficult course of 132 miles in the desert in totally autonomous mode. The next [DARPA](#) challenge in 2007 will see multiple vehicles operating simultaneously in an urban environment; this challenge will certainly bring products based on autonomous vehicle techniques closer to the market [51.76].

51.7 Future Trends and Prospects

Our cars, trucks, and buses are inevitably getting smarter. The future path is not completely known, and is likely to vary in different locations.

Several trends are clear, however. As sensor-equipped vehicles become more common, there will be increasing opportunities to build sensor-friendly technologies into roadways and into vehicles. It is straightforward to reduce the amount of radar clutter on roadsides by replacing large metal signs, which are excellent radar reflectors, with nonreflective signs made of plastic or composites. It will be desirable to make overpasses with lower radar reflectivities, either by using different materials, by coating steel structures with radar-absorbing material, or by proper geometry design. At the same time, it will be desirable to increase the radar cross-section of small vehicles, such as motorcycles, to make them easier for radar-equipped cars to detect. Radar-reflecting licence plates, for in-

stance, would make smaller vehicles stand out more clearly.

The next steps may be to deploy systems that support the active transmission information from infrastructure to vehicle and vehicle to vehicle. Dedicated short-range communications ([DSRC](#)) is already in wide use in smart toll collection. It is easy to imagine [DSRC](#)-equipped vehicles receiving alerts from roadside sensors of upcoming fog, or ice, or congestion. Intelligent vehicles that detect unusual roadway conditions (via radar or lidar, slip detectors, or driver actions such as hard braking) could broadcast that information to all nearby smart vehicles. As market penetration of equipped vehicles increases, the value of such ad hoc communications nets will go up dramatically.

Vehicle-to-vehicle communications need not be only via radio. Light-emitting diode (LED) brake lights can

be pulsed at kilohertz frequencies, well above the bandwidth at which human eyes can detect flicker, but easily received by following vehicles. This would be a straight-

forward way for a vehicle to tell other vehicles about the onset of hard braking, or other unusual roadway conditions.

51.8 Conclusions and Further Reading

The general public will become increasingly accustomed to intelligent systems – sensors and communications on vehicles. The only showstopper in the large-scale deployment of robotics technologies in the automobile field is the ability of the industry to deliver the technologies with total safety, which focus attention on the problems of failsafe systems and their certification. The rail and aerospace industries have solved these problems but in a very different environment. In these industries, the cost of safety for each vehicle can be much higher than in a car or a bus and the operational environment is also quite different, with professionals operating and maintaining the system. Manufacturers will slowly develop experience in reliability and cost engineering, and governments will gradually work out liability issues. For these reasons, systems that put the driver in the loop of new automation technologies will be the initial focus of development.

This technical progress still will not answer societal questions. Will smart vehicles make public transportation more efficient and desirable? Or will they make it easier for people to use their personal vehicles, since driving will be less stressful? The nature of transport, and its effect on urban areas, is not easily predictable. Intelligent transportation systems of the future will provide the tools to shape these discussions; society will have to choose its future from an increasingly rich set of alternatives.

For further reading on this topic there are a number of sources that can be recommended:

- intelligent vehicles survey text [51.77]
- intelligent transport systems journals [51.78–80]
- annual conferences [51.81–84]
- government resources [51.85–89]
- magazines [51.90]

References

- | | |
|---|---|
| <p>51.1 E.D. Dickmanns: The development of machine vision for road vehicles in the last decade, Proc. IEEE Intelligent Vehicles Symposium 2002, Vol. 1, Paris (2002) pp. 268–281</p> <p>51.2 A. Broggi, M. Bertozzi, A. Fascioli, G. Conte: <i>Automatic Vehicle Guidance: The Experience of the ARGO Autonomous Vehicle</i> (World Scientific, Singapore 1999)</p> <p>51.3 C.E. Thorpe (Ed.): <i>Vision and Navigation: The Carnegie Mellon Navlab</i> (Kluwer Academic, Dordrecht 1990)</p> <p>51.4 D. Pomerleau, T. Jochem: Rapidly adapting machine vision for automated vehicle steering, IEEE Expert, 11(2) 19–27 (1996), Special Issue on Intelligent System and their Applications</p> <p>51.5 US Department of Transport: Intelligent Transportation Systems, Saving Lives Through Advanced Vehicle Safety Technology: Intelligent Vehicle Initiative Final Report (2005) Publication No. FHWA-JPO-05-057. http://www.its.dot.gov/ivi/ivi.htm</p> <p>51.6 K. Iagnemma, M. Buehler (Eds.): Special Issues on the DARPA Grand Challenge, J. Field Robot. 23(8,9) (2006), www.darpa.mil/grandchallenge</p> | <p>51.7 L. Vlacic, M. Parent, F. Harashima: <i>EDS Intelligent Vehicle Technologies, Theory and Applications</i> (Butterworth-Heinemann, Oxford 2001)</p> <p>51.8 Micron: Supplier of imaging sensors including infra-red (Micron, Boise 2007), www.micron.com</p> <p>51.9 Valeo: Supplier of automotive technology – including ultrasonic parking assistance systems www.valeo.com</p> <p>51.10 Hella: Supplier of automotive 24-GHz radar sensors www.hella.com</p> <p>51.11 Ibeo: Supplier of automotive laser sensors www.ibeo-as.com</p> <p>51.12 Mobileye: Supplier of automotive computer vision sensors www.mobileye.com</p> <p>51.13 Seeing Machines: Supplier of driver state vision systems www.seeingmachines.com</p> <p>51.14 Leearmstrong: Support site for Dedicated Short Range Communications (DSRC) discussions www.leearmstrong.com</p> <p>51.15 Sae: ISO/TC204 Transport Information and Control Systems (TICS), Working Group 16, Wide Area Communications/Protocols and Interfaces http://www.sae.org/technicalcommittees/tc204wg16.htm</p> |
|---|---|

- 51.16 T. Ernst: The information technology era of the vehicular industry, ACM SIGCOMM Comput. Commun. Rev. (CCR) **36**(2), 49–52 (2006), <http://www.acm.org/sigsigcomm/ccr/archive/2006/april/p49-ernst.pdf>
- 51.17 CALM working group producing standards and specifications <http://www.calm.hu> and <http://www.calmforum.com>
- 51.18 CVIS: *Cooperative Vehicle-Infrastructure Systems* (ERTICO, Brussels 2007), http://www.ertico.com/en/activities/efficiency_environment/cvis.htm
- 51.19 ISOC: *Internet Engineering Task Force* (ISOC, Sterling 2007), <http://www.ietf.org>
- 51.20 Sigmobile: Annual series of workshops. Lastest iteration The Third ACM International Workshop on Vehicular ad hoc Networks (VANET 2006), <http://www.sigmobile.org/workshops/vanet2006/>
- 51.21 Trimble: Manufacturer of GPS navigation systems www.trimble.com
- 51.22 Navman: Manufacturer of in-car navigation systems www.navman.com
- 51.23 Ertico: actMAP an EU project for dynamic map updating (ERTICO, Brussels 2007), www.ertico.com
- 51.24 S. Rogers, W. Zhang: Development and evaluation of a curve rollover warning system for trucks, IEEE IV2003 Intelligent Vehicles Symposium, Columbus (2003) pp. 294–297
- 51.25 D. Vaughan: Vehicle speed control based on GPS/MAP matching of posted speeds, US Patent 5485161, Trimble Navigation Limited
- 51.26 IN-ARTE: An EU project for combining maps and sensors safety applications (Frauenhofer IAO, Stuttgart 2007), <http://www.iao.fraunhofer.de/Projects/IN-ARTE/>
- 51.27 D.M. Gavrilu, U. Franke, C. Wöhler, S. Götzig: Real-time vision for intelligent vehicles, IEEE Instr. Measure. Mag. **4**(2), 22–27 (2001)
- 51.28 Japanese Advanced Safety Vehicle Project (Road Transportation Bureau, Tokyo 2007), http://www.mlit.go.jp/jidosha/anzen/asv/ASV_English/ASV3.htm
- 51.29 Iteris: Supplier of lane tracking systems www.iteris.com
- 51.30 N.E. Apostoloff, A. Zelinsky: Robust vision based lane tracking using multiple cues and particle filtering. Proc. IEEE Intelligent Vehicles Symposium, Columbus (2003) pp. 558–563
- 51.31 D.M. Gavrilu, V. Philomin: Real-time object detection for ‘smart’ vehicles. Proc. Seventh IEEE International Conference on Computer Vision, Kerkyra, Vol.1 (1999) pp. 87–93
- 51.32 G. Piccioli, E. De Micheli, P. Parodi, M. Campani: Robust method for road sign detection and recognition, Image Vis. Comput. **14**(3), 209–223 (1996)
- 51.33 C. Bahlmann, Y. Zhu, R. Visvanathan, M. Pellkofer, T. Koehler: A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. Proc. IEEE Intelligent Vehicles Symposium (2005) pp. 255–260
- 51.34 U. Franke, A. Joos: Real-time stereo vision for urban traffic scene understanding. Proc. IEEE Intelligent Vehicles Symposium, Dearborn (2000) pp. 273–278
- 51.35 Nissan safety vehicle that interacts with infrastructure (ERTICO, Brussels 2007), http://www.ertico.com/en/news_and_events/ertico_newsroom/nissan.htm
- 51.36 N. Hautiere, R. Labayrade, D. Aubert: Estimation of the Visibility Distance by Stereovision: A Generic Approach, IEICE Trans. Inform. Syst. **E89-D**(7), 2084–2091 (2006)
- 51.37 R. Labayrade, D. Aubert, J.P. Tarel: Real Time Obstacle Detection in Stereo Vision on non Flat Road Geometry through ‘V-Disparity’ Representation. Proc. IEEE Intelligent Vehicles Symposium, Paris, Vol.2 (2002) pp. 646–651
- 51.38 Z. Sun, G. Bebis, R. Miller: On-road vehicle detection: A review, IEEE Trans. Pattern Anal. Machine Intell. (PAMI) **28**(5), 694–711 (2006)
- 51.39 E. Segawa, M. Shiohara, S. Sasaki, N. Hashiguchi, T. Takashima, M. Tohno: Preceding vehicle detection using stereo images and non-scanning millimeter-wave radar, IEICE Trans. Inform. Syst. **E89-D**(7), 2101–2108 (2006)
- 51.40 T. Kato, Y. Ninomiya, I. Masaki: An obstacle detection method by fusion of radar and motion stereo, IEEE Trans. Intell. Transport. Syst. **3**(3), 182–188 (2002)
- 51.41 M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, T. Poggio: Pedestrian detection using wavelet templates, Computer Vision and Pattern Recognition, San Juan (1997) pp. 193–199
- 51.42 L. Zhao, C.E. Thorpe: Stereo- and neural network-based pedestrian detection, IEEE Trans. Intell. Transport. Syst. **01**(3), 148–154 (2000)
- 51.43 D. Pomerleau: Ralph: Rapidly adapting lateral position handler. In Proc. IEEE Symposium on Intelligent Vehicles, Detroit (1995) pp. 506–511
- 51.44 US DOT: Intelligent Vehicle Initiative Program (US DOT IVI, Washington 2007), www.its.dot.gov/ivi/ivi.htm
- 51.45 Roadranger: Eaton-Vorad Road Ranger BlindSpotter product www.roadranger.com
- 51.46 L. Matuszyk, A. Zelinsky, L. Nilsson, M. Rilbe: Stereo panoramic vision for monitoring vehicle blind-spots. Proc. IEEE Intelligent Vehicles Symposium, Parma (2004) pp. 31–36
- 51.47 Motiontrends: Volvo S80 vehicle with advanced driver assistance systems (Motion Trends, Toulouse 2006), www.motiontrends.com/2006csm/m03/volvo/volvo_s80_safety_systems.shtml
- 51.48 Visteon: Automotive supplier – including night vision www.visteon.com
- 51.49 T.A. Williamson: *A High-Performance Stereo Vision System for Obstacle Detection*, Tech. Rep. CMU-

- RI-TR-98-24 Robotics Institute (Carnegie Mellon University, Pittsburgh 1998)
- 51.50 E. Li: Millimeter-wave polarimetric radar system as an advanced vehicle control and warning sensor. Ph.D. Thesis (University of Michigan, Michigan 1998)
- 51.51 J.A. Hancock: Laser Intensity Based Obstacle Detection and Tracking. Ph.D. Thesis (Carnegie Mellon University, Pittsburgh 1999)
- 51.52 W.D. Jones: Keeping cars from crashing, *IEEE Spectrum* **38**(9), 40–45 (2001)
- 51.53 Delphi: Supplier of automotive technology – including stop-go adaptive cruise control systems www.delphi.com
- 51.54 I.E. Paromtchik, C. Laugier: Motion generation and control for parking an autonomous vehicle. *Proc. International Conference on Robotics and Automation*, Orlando, Vol. 4 (1996) pp. 3117–3122
- 51.55 Toyota: Toyota parking assistance system www.toyota.co.jp
- 51.56 V.R. Vuchic: O-Bahn: Description and evaluation of a new concept, *Advances in Bus Service Planning Practices*, Transportation Research Record No. 1011, (Transportation Research Board, Washington DC 1985)
- 51.57 S. Ishida, J.E. Gayko: Development, Evaluation and Introduction of a Lane Keeping Assistance System. *Proc. IEEE Intelligent Vehicles Symposium*, Parma (2004) pp. 943–944
- 51.58 Siemens: Siemens-VDO lane changing assist www.siemensvdo.com
- 51.59 J. Treat, N. Tumbas, S. McDonald, D. Shinar, R. Hume, R. Mayer, R. Stansifer, N. Castellan: Tri-Level study of the causes of traffic accidents: Final report – Executive summary. Tech. Rep. DOT-HS-034-3-535-79-TAC(S) (Institute for Research in Public Safety, Bloomington 1979)
- 51.60 M. Carmen del Rio, J. Gomez, M. Sancho, F.J. Alvarez: Alcohol, illicit drugs and medicinal drugs in fatally injured drivers in Spain between 1991 and 2000, *Forensic Sci. Int.* **127**(1–2), 63–70(8) (2002)
- 51.61 N.L. Haworth, T.J. Triggs, E.M. Grey: *Driver Fatigue: Concepts, Measurement and Crash Countermeasures*. Transport and Communications, Federal Office of Road Safety, Human Factors Group, Department of Psychology, Monash University (1988)
- 51.62 N.L. Howarth, C.J. Heffernan, E.J. Horne: Fatigue in truck accidents. Tech. Rep. 3, Monash University Accident Research Centre (1989)
- 51.63 W.W. Wierwille, L.A. Ellsworth: Evaluation of driver drowsiness by trained raters, *Accident Anal Prevent.* **26**(5), 571–581 (1994)
- 51.64 J. Stutts, D. Reinfurt, L. Staplin, E. Rodgman: The role of driver distraction in traffic crashes. Tech. Rep., Foundation for Traffic Safety (AAA Washington 2001)
- 51.65 T. Victor: Keeping Eye and Mind on the Road. Ph.D. Thesis (Uppsala University, Sweden, 2005)
- 51.66 J.E. Santos Conde, M.A. Hillebrand, A. Teuner, N. Stevanovic, U. Lurgel, B.J. Hosticka: A smart airbag solution based on a high speed CMOS camera system. *Internet Conference on Image Processing*, Kobe, Vol. 3 (1999) pp. 930–934
- 51.67 Volvo: Future safety features of Volvo vehicles (Volvo Trade Corp., Göteborg 2002), <http://www.volvo.com/NR/rdonlyres/6114A843-D711-4A5E-9D56-9ACD501A2289/0/MRilbe.pdf>
- 51.68 M. Schulze, J. Irion, T. Mäkinen, M. Flament: Accidentology as a basis for requirements and system architecture of preventive safety applications, 10th International Forum on Advanced Microsystems for Automotive Applications (AAA 2006), Berlin (2006), http://ip-prevent.org/en/public_documents/publications/papers_presentations
- 51.69 X.Y. Lu, H.S. Tan, S.E. Shladover, J.K. Heidrick: Implementation and comparison of nonlinear longitudinal controllers for car platooning, *AVEC 2000*, 5th International Symposium on Advanced Vehicle Control. Ann Arbor (2000)
- 51.70 Cybercar: Cybercar – Automated Bus Rapid Transit vehicle www.cybercars.org
- 51.71 European Commission: Intelligent Car Initiative. Communication from the European Commission. COM-2006-59 (European Commission, Brussels 2006), <http://europa.eu.int>
- 51.72 World Carshare Consortium, Paris (2007), http://www.ecoplan.org/carshare/cs_index.htm
- 51.73 R. Cervero: Creating a linear city with a surface metro: The story of Curitiba, *Institute of Urban and Regional Development*, University of California, Berkeley, IURD Working Paper 643 (1995)
- 51.74 H.S. Jacob Taso, J.L. Botha: Definition and Evaluation of Bus and Truck Automation Operations Concepts. Path Rep. UCB-ITS-PRR-2002-08. University of California (2002)
- 51.75 Phileas Bus, <http://www.phileas.nl>
- 51.76 2007 Darpa Grand Challenge (Department of Defense, Washington 2007), <http://www.darpa.mil/grandchallenge/index.asp>
- 51.77 R. Bishop: *Intelligent Vehicle Technology and Trends* (Artech House, Boston 2005)
- 51.78 IEEE Transactions on Intelligent Transportation Systems (ITS), Journal published by IEEE
- 51.79 IEEE Transactions on Vehicular Technology, Journal published by IEEE
- 51.80 Transportation Research PART B: Methodology, Journal by Elsevier
- 51.81 Intelligent Vehicles Symposium. Annual conference Sponsored by IEEE
- 51.82 Intelligent Transportation Systems Conference. Annual conference Sponsored by IEEE
- 51.83 Vehicular Technology Conference Annual conference. Sponsored by IEEE

51.84

World Congress on Intelligent Transportation Systems. Annual conference. Sponsored by ITS Societies

51.85

US Department of Transportation, ITS Joint Program Office (DOT, Washington 2007), <http://www.its.dot.gov/>

51.86

European Union Research on Transportation (European Commission, Brussels 2007), <http://ec.europa.eu/research/transport/>

51.87

Advanced Driver Assistance Systems in Europe (ADASE Daimler Chrysler, Brussels 2007), <http://www.adase2.net/>

51.88

Japan Ministry of Land, Infrastructure and Transport, Road Bureau ITS Program (MLIT, Tokyo 2007), <http://www.its.go.jp/ITS/>

51.89

INVENT: A German program combining intelligent vehicles and infrastructure (BMFT, Berlin 2007), www.invent-online.de/en/projects.html

51.90

ITS Magazine, published by IEEE

52. Medical Robotics and Computer-Integrated Surgery

Russell H. Taylor, Arianna Menciassi, Gabor Fichtinger, Paolo Dario

The growth of medical robotics since the mid-1980s has been striking. From a few initial efforts in stereotactic brain surgery, orthopaedics, endoscopic surgery, microsurgery, and other areas, the field has expanded to include commercially marketed, clinically deployed systems, and a robust and exponentially expanding research community. This chapter will discuss some major themes and illustrate them with examples from current and past research. Further reading providing a more comprehensive review of this rapidly expanding field is suggested in Sect. 52.4.

Medical robots may be classified in many ways: by manipulator design (e.g., kinematics, actuation); by level of autonomy (e.g., preprogrammed versus teleoperation versus constrained cooperative control), by targeted anatomy or technique (e.g., cardiac, intravascular, percutaneous, laparoscopic, microsurgical); or intended operating environment (e.g., in-scanner, conventional operating room). In this chapter, we have chosen to focus on the role of medical robots within the context of larger computer-integrated systems including presurgical planning, intraoperative execution, and postoperative assessment and follow-up.

First, we introduce basic concepts of computer-integrated surgery, discuss critical factors affecting the eventual deployment and acceptance of medical robots, and introduce the basic system paradigms of surgical computer-assisted planning, registration, execution, monitoring, and assessment (CAD/CAM) and surgical assistance. In subsequent sections, we provide an overview of the technology of medical robot systems and discuss examples of our basic system paradigms, with brief additional discussion topics of remote telesurgery and robotic surgical simulators. We conclude with some thoughts on future research directions and provide suggested further reading.

52.1 Core Concepts	1200
52.1.1 Medical Robotics, Computer-Integrated Surgery, and Closed-Loop Interventions	1200
52.1.2 Factors Affecting the Acceptance of Medical Robots	1200
52.1.3 Medical Robotics System Paradigms: Surgical CAD/CAM and Surgical Assistance	1202
52.2 Technology	1204
52.2.1 Mechanical Design Considerations	1204
52.2.2 Control Paradigms	1205
52.2.3 Virtual Fixtures and Human-Machine Cooperative Systems	1206
52.2.4 Safety and Sterility	1207
52.2.5 Imaging and Modeling of Patients	1208
52.2.6 Registration	1208
52.3 Systems, Research Areas, and Applications	1209
52.3.1 Nonrobotic Computer-Assisted Surgery: Navigation and Image Overlay Devices	1209
52.3.2 Orthopaedic Systems	1209
52.3.3 Percutaneous Needle Placement Systems	1210
52.3.4 Telesurgical Systems	1212
52.3.5 Microsurgery Systems	1213
52.3.6 Endoluminal Robots	1213
52.3.7 Sensorized Instruments and Haptic Feedback	1214
52.3.8 Surgical Simulators and Telerobotic Systems for Training	1215
52.3.9 Other Applications and Research Areas	1216
52.4 Conclusion and Future Directions	1217
References	1218

52.1 Core Concepts

52.1.1 Medical Robotics, Computer-Integrated Surgery, and Closed-Loop Interventions

A fundamental property of robotic systems is their ability to couple complex information to physical action in order to perform a useful task. This ability to replace, supplement, or transcend human performance has had a profound influence on many fields of our society, including industrial production, exploration, quality control, and laboratory processes. Although robots have often been first introduced to automate or improve discrete processes such as welding or test probe placement or to provide access to environments where humans cannot safely go, their greater long-term impact has often come indirectly as essential enablers of computer integration of entire production or service processes.

Medical robots have a similar potential to fundamentally change surgery and interventional medicine as part of a broader, information-intensive environment that exploits the complementary strengths of humans and computer-based technology. The robots may be thought of as information-driven surgical tools that enable human surgeons to treat individual patients with greater safety, improved efficacy, and reduced morbidity than would otherwise be possible. Further, the consistency and information infrastructure associated with medical robotic and computer-assisted surgery systems have the potential to make *computer-integrated surgery* as important to health care as computer-integrated manufacturing is to industrial production.

Figure 52.1 illustrates this view of computer-integrated surgery (CIS). The process starts with information about the patient, which can include medical images [computed tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), etc.], lab test results, and other information. This patient-specific information is combined with statistical information about human anatomy, physiology, and disease to produce a comprehensive computer representation of the patient, which can then be used to produce an optimized interventional plan. In the operating room, the preoperative patient model and plan must be *registered* to the actual patient. Typically, this is done by identifying corresponding landmarks or structures on the preoperative model and the patient, either by means of additional imaging (X-ray, ultrasound, video), by the use of a tracked pointing device, or by the robot itself. If the patient’s anatomy has changed, then the model and plan are updated appropriately, and the planned procedure is carried out with assistance of the robot. As the intervention continues, additional imaging or other sensing is used to monitor the progress of the procedure, to update the patient model, and to verify that the planned procedure has been successfully executed. After the procedure is complete, further imaging, modeling, and computer-assisted assessment is performed for patient follow-up and to plan subsequent interventions, if any should be required. Further, all the patient-specific data generated during the planning, execution, and follow-up phases can be retained. These data can subsequently be analyzed statistically to improve the rules and methods used to plan future procedures.

52.1.2 Factors Affecting the Acceptance of Medical Robots

Medical robotics is ultimately an application-driven research field. Although the development of medical robotic systems requires significant innovation and can lead to very real, fundamental advances in technology, medical robots must provide measurable and significant advantages if they are to be widely accepted and deployed. The situation is complicated by the fact that these advantages are often difficult to measure, can take an extended period to assess, and may be of varying importance to different groups. Table 52.1 lists some of the more important factors that researchers contemplating the development of a new medical robot system should consider in assessing their proposed approach.

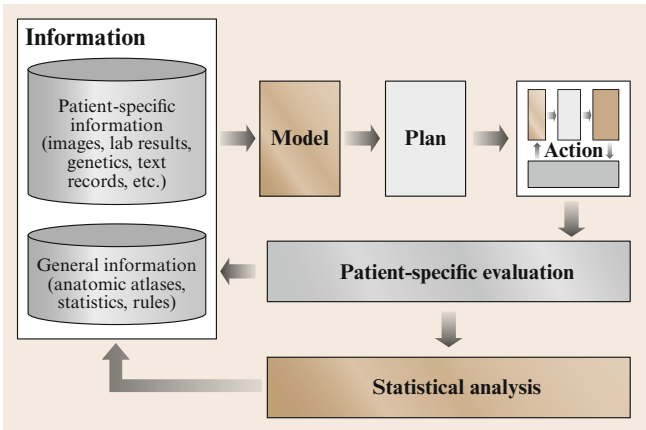


Fig. 52.1 Fundamental information flow in computer-integrated surgery

Table 52.1 Assessment factors for medical robots or computer-integrated surgery systems [52.1]

Assessment factor	Important to whom	Assessment method	Summary of key leverage
New treatment options	Clinical researchers, patients	Clinical and trials preclinical	Transcend human sensory-motor limits (e.g., in microsurgery). Enable less invasive procedures with real-time image feedback (e.g., fluoroscopic or MRI -guided liver or prostate therapy). Speed up clinical research through greater consistency and data gathering
Quality	Surgeons, patients	Clinician judgment; revision rates	Significantly improve the quality of surgical technique (e.g., in microvascular anastomosis), thus improving results and reducing the need for revision surgery
Time and cost	Surgeons, hospitals, insurers	Hours, hospital charges	Speed operating room (OR) time for some interventions. Reduce costs from healing time and revision surgery. Provide effective intervention to treat patient condition
Less invasiveness	Surgeons, patients	Qualitative judgment; recovery times	Provide crucial information and feedback needed to reduce the invasiveness of surgical procedures, thus reducing infection risk, recovery times, and costs (e.g., percutaneous spine surgery)
Safety	Surgeons, patients	Complication and revision surgery rates	Reduce surgical complications and errors, again lowering costs, improving outcomes and shortening hospital stays (e.g., robotic total hip replacement (THR), steady-hand brain surgery)
Real-time feedback	Surgeons	Qualitative assessment, quantitative comparison of plan to observation, revision surgery rates	Integrate preoperative models and intraoperative images to give surgeon timely and accurate information about the patient and intervention (e.g., fluoroscopic X-rays without surgeon exposure, percutaneous therapy in conventional MRI scanners). Assure that the planned intervention has in fact been accomplished
Accuracy or precision	Surgeons	Quantitative comparison of plan to actual	Significantly improve the accuracy of therapy dose pattern delivery and tissue manipulation tasks (e.g., solid organ therapy, microsurgery, robotic bone machining)
Enhanced documentation and follow-up	Surgeons, clinical researchers	Databases, anatomical atlases, images, and clinical observations	Exploit CIS systems' ability to log more varied and detailed information about each surgical case than is practical in conventional manual surgery. Over time, this ability, coupled with CIS systems' consistency, has the potential to significantly improve surgical practice and shorten research trials

Broadly, the advantages offered by medical robots may be grouped into three areas. The first is the potential of a medical robot to *significantly improve surgeons' technical capability* to perform procedures by exploiting the complementary strengths of humans and robots summarized in Table 52.2. Medical robots can be constructed to be more precise and geometrically accurate than an unaided human. They can operate in hostile radiological environments and can provide great dexterity for minimally invasive procedures inside the patient's body.

These capabilities can both enhance the ability of an *average* surgeon to perform procedures that only a few exceptionally gifted surgeons can perform unassisted and can also make it possible to perform interventions that would otherwise be completely infeasible.

A second, closely related capability is the potential of medical robots to *promote surgical safety* both by improving a surgeon's technical performance and by means of active assists such as *no-fly zones* or *virtual fixtures* (Sect. 52.2.3) to prevent surgical instruments

Table 52.2 Complementary strengths of human surgeons and robots [52.1]

	Strengths	Limitations
Humans	Excellent judgment Excellent hand–eye coordination Excellent dexterity (at natural <i>human</i> scale) Able to integrate and act on multiple information sources Easily trained Versatile and able to improvise	Prone to fatigue and inattention Limited fine motion control due to tremor Limited manipulation ability and dexterity outside natural scale Cannot see through tissue Bulky end-effectors (hands) Limited geometric accuracy Hard to keep sterile Affected by radiation, infection
Robots	Excellent geometric accuracy Untiring and stable Immune to ionizing radiation Can be designed to operate at many different scales of motion and payload Able to integrate multiple sources of numerical and sensor data	Poor judgment Hard to adapt to new situations Limited dexterity Limited hand–eye coordination Limited haptic sensing (today) Limited ability to integrate and interpret complex information

from causing unintentional damage to delicate structures. Furthermore, the integration of medical robots within the information infrastructure of a larger CIS system can provide the surgeon with significantly improved monitoring and online decision supports, thus further improving safety.

A third advantage is the inherent ability of medical robots and CIS systems to *promote consistency* while *capturing detailed online information* for every procedure. Consistent execution (e.g., in spacing and tension-

ing of sutures or in placing of components in joint reconstructions) is itself an important quality factor. If saved and routinely analyzed, the *flight data recorder* information inherently available with a medical robot can be used both in *morbidity and mortality* assessments of serious surgical incidents and, potentially, in statistical analyses examining many cases to develop better surgical plans. Furthermore, such data can provide valuable input for surgical simulators, as well as a database for developing skill assessment and certification tools for surgeons.

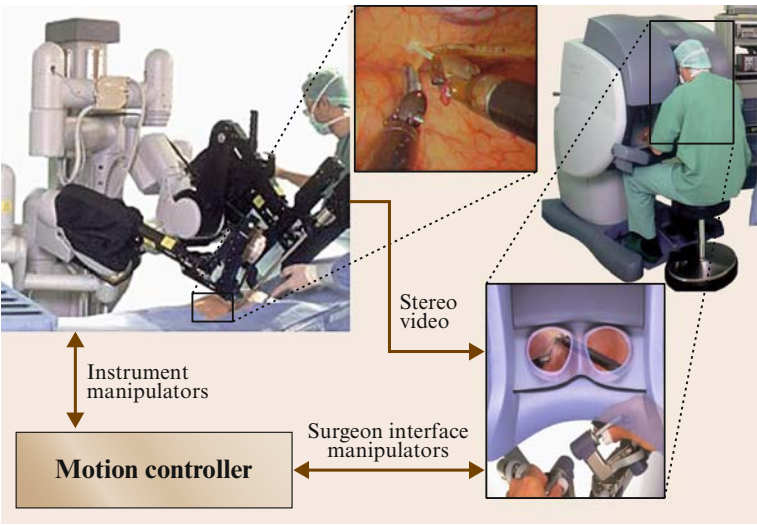


Fig. 52.2 The da Vinci telesurgical robot [52.2] extends a surgeon’s capabilities by providing the immediacy and dexterity of *open* surgery in a minimally invasive surgical environment. (Photos: Intuitive Surgical, Sunnyvale)

52.1.3 Medical Robotics System Paradigms: Surgical CAD/CAM and Surgical Assistance

We call the process of computer-assisted planning, registration, execution, monitoring, and assessment *surgical CAD/CAM*, emphasizing the analogy to manufacturing CAD/CAM. Just as with manufacturing, robots can be critical in this CAD/CAM process by enhancing the surgeon's ability to execute surgical plans. The specific role played by the robot depends somewhat on the application, but current systems tend to exploit the geometric accuracy of the robot and/or its ability to function concurrently with X-ray or other imaging devices. Typical examples include radiation therapy delivery robots such as Accuray's CyberKnife [52.5] (Accuray, Inc., Sunnyvale, CA.), shaping of bone in

orthopaedic joint reconstructions (discussed further in Sect. 52.3.2) and image-guided placement of therapy needles (Sect. 52.3.3).

Surgery is often highly interactive; many decisions are made by the surgeon in the operating room and executed immediately, usually with direct visual or haptic feedback. Generally, the goal of surgical robotics is not to replace the surgeon so much as to improve his or her ability to treat the patient. The robot is thus a computer-controlled surgical tool in which control of the robot is often shared in one way or another between the human surgeon and a computer. We thus often speak of medical robots as *surgical assistants*.

Broadly, robotic surgical assistants may be broken into two subcategories. The first category, *surgeon extender robots*, manipulate surgical instruments under the direct control of the surgeon, usually through a teleop-

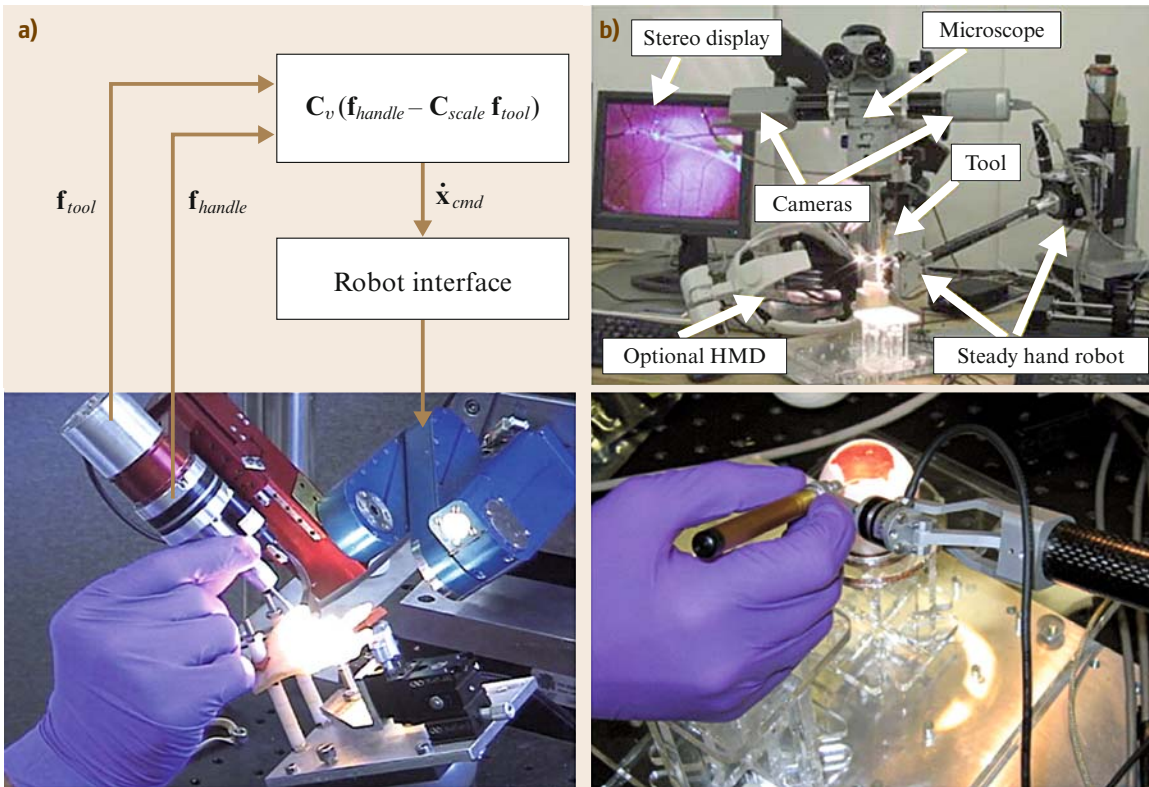


Fig. 52.3a,b The Johns Hopkins *Steady Hand* microsurgical robot [52.3,4] extends a surgeon's capabilities by providing the ability to manipulate surgical instruments with very high precision while still exploiting the surgeon's natural hand-eye coordination. **(a)** The basic paradigm of hands-on compliant guiding. The commanded velocity of the robot is proportional to a scaled difference between the forces exerted by the surgeon on the tool handle and (optionally) sensed tool-to-tissue forces. **(b)** A more recent version of the Steady Hand robot currently being used for experiments in microcannulation of 100 μm blood vessels

eration or *hands-on* cooperative control interface. The primary value of these systems is that they can overcome some of the perception and manipulation limitations of the surgeon. Examples include the ability to manipulate surgical instruments with superhuman precision by eliminating hand tremor, the ability to perform highly dexterous tasks inside the patient's body, or the ability to perform surgery on a patient who is physically remote from the surgeon. Although setup time is still a serious concern with most surgeon extender systems, the greater ease of manipulation that such systems offer has the potential to reduce operative times. One widely deployed example of a surgeon extender is the daVinci system [52.2] (Intuitive Surgical Systems, Sunnyvale, CA) shown in Fig. 52.2. Other examples include the Sensei catheter system [52.6] (Hansen Medical Systems, Mountain View, CA.) and the experimental Johns Hopkins University (JHU) *Steady Hand* microsurgery robot shown in Fig. 52.3. Further examples are discussed in Sect. 52.3.

A second category, *auxiliary surgical support robots*, generally work alongside the surgeon and perform such routine tasks as tissue retraction, limb

positioning, or endoscope holding. One primary advantage of such systems is their potential to reduce the number of people required in the operating room, although that advantage can only be achieved if *all* the tasks routinely performed by an assisting individual can be automated. Other advantages can include improved task performance (e.g., a steadier endoscopic view), safety (e.g., elimination of excessive retraction forces), or simply giving the surgeon a greater feeling of control over the procedure. One of the key challenges in these systems is providing the required assistance without posing an undue burden on the surgeon's attention. A variety of control interfaces are common, including joysticks, head tracking, voice recognition systems, and visual tracking of the surgeon and surgical instruments, for example, the Aesop endoscope positioner [52.7] used both a foot-actuated joystick and a very effective voice recognition system. Again, further examples are discussed in Sect. 52.3.

It is important to realize that *surgical CAD/CAM* and *surgical assistance* are complementary concepts. They are not at all incompatible, and many systems have aspects of both.

52.2 Technology

52.2.1 Mechanical Design Considerations

The mechanical design of a surgical robot depends crucially on its intended application. For example, robots with high precision, stiffness and (possibly) limited dexterity are often very suitable for orthopaedic bone shaping or stereotactic needle placement, and medical robots for these applications [52.8–11] frequently have high gear ratios and consequently, low back-drivability, high stiffness, and low speed. On the other hand, robots for complex, minimally invasive surgery (MIS) on soft tissues require compactness, dexterity, and responsiveness. These systems [52.2, 12] frequently have relatively high speed, low stiffness, and highly back-drivable mechanisms.

Many early medical robots [52.8, 11, 13] were essentially modified industrial robots. This approach has many advantages, including low cost, high reliability, and shortened development times. If suitable modifications are made to ensure safety and sterility, such systems can be very successful clinically [52.9], and they can also be invaluable for rapid prototyping and research use.

However, the specialized requirements of surgical applications have tended to encourage more specialized

designs. For example, laparoscopic surgery and percutaneous needle placement procedures typically involve the passage or manipulation of instruments about a common entry point into the patient's body. There are two basic design approaches. The first approach uses a passive wrist to allow the instrument to pivot about the insertion point and has been used in the commercial Aesop and Zeus robots [52.12, 14] as well as several research systems. The second approach mechanically constrains the motion of the surgical tool to rotate about a *remote center of motion (RCM)* distal to the robot's structure. In surgery, the robot is positioned so that the RCM point coincides with the entry point into the patient's body. This approach has been used by the commercially developed daVinci robot [52.2], as well as by numerous research groups, using a variety of kinematic designs [52.15–17].

The emergence of minimally invasive surgery has created a need for robotic systems that can provide high degrees of dexterity in very constrained spaces inside the patient's body, and at smaller and smaller scales. Figure 52.4 shows several typical examples of current approaches. One common response has been to develop cable-actuated wrists [52.2]. However, a num-

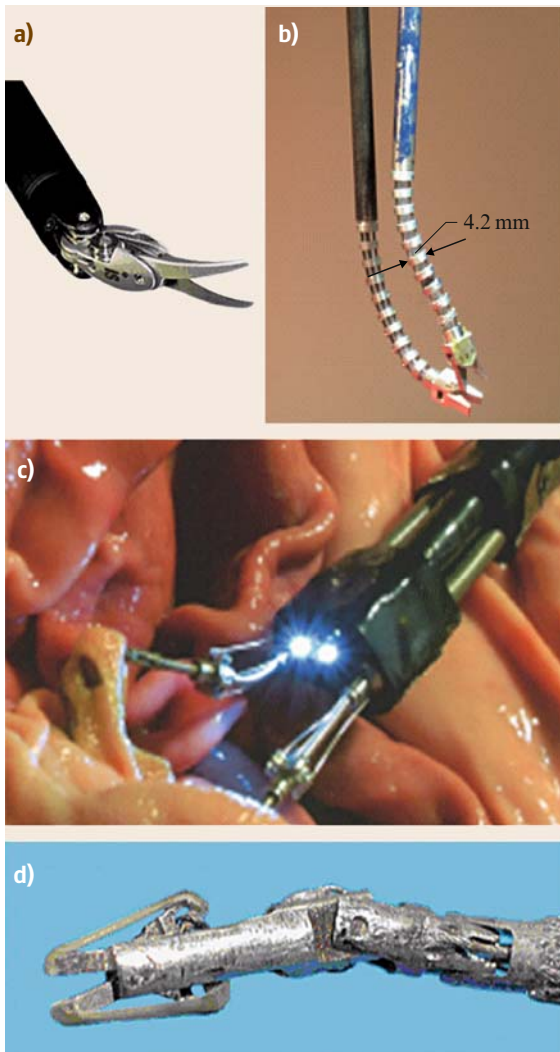


Fig. 52.4a–d Dexterity enhancement inside a patient's body: (a) The daVinci wrist with a typical surgical instrument (here, scissors) [52.2]; (b) The end-effectors of the JHU/Columbia snake telesurgical system [52.18]; (c) Two-handed manipulation system for use in endogastrectomy [52.26]; (d) five-degree-of-freedom 3 mm wrist and gripper [52.27] for microsurgery in deep and narrow spaces

ber of investigators have investigated other approaches, including bending structural elements [52.18], shape-memory alloy actuators [52.19, 20], microhydraulic systems [52.21], and electroactive polymers [52.22]. Similarly, the problem of providing access to surgical sites inside the body has led several groups to develop

semiautonomously moving robots for epicardial [52.23] or endoluminal applications [52.24, 25].

Although most surgical robots are mounted to the surgical table, to the operating room ceiling, or to the floor, there has been some interest in developing systems that directly attach to the patient [52.28, 29]. The main advantage of this approach is that the relative position of the robot and patient is unaffected if the patient moves. The challenges are that the robot must be smaller and that relatively noninvasive means for mounting it must be developed.

Finally, robotic systems intended for use in specific imaging environments pose additional design challenges. First, there is the geometric constraint that the robot (or at least its end-effector) must fit within the scanner along with the patient. Second, the robot's mechanical structure and actuators must not interfere with the image formation process. In the case of X-ray and CT, satisfying these constraints is relatively straightforward. The constraints for MRI are more challenging [52.30].

52.2.2 Control Paradigms

Surgical robots assist surgeons in treating patients by moving surgical instruments, sensors, or other devices in relation to the patient. Generally, these motions are controlled by the surgeon in one of three ways:

- *Preprogrammed, semi-autonomous motion:* The desired behavior of the robot's tools is specified interactively by the surgeon, usually based on medical images. The computer fills in the details and obtains the surgeon's concurrence before the robot is moved. Examples include the selection of needle target and insertion points for percutaneous therapy and tool cutter paths for orthopaedic bone machining.
- *Teleoperator control:* The surgeon specifies the desired motions directly through a separate human interface device and the robot moves immediately. Examples include common telesurgery systems such as the daVinci [52.2]. Although physical master manipulators are the most common input devices, other human interfaces are also used, notably voice control [52.12].
- *Hands-on compliant control:* The surgeon grasps the surgical tool held by the robot or a control handle on the robot's end-effector. A force sensor senses the direction that the surgeon wishes to move the tool and the computer moves the robot to comply. Early experiences with Robodoc [52.8] and other

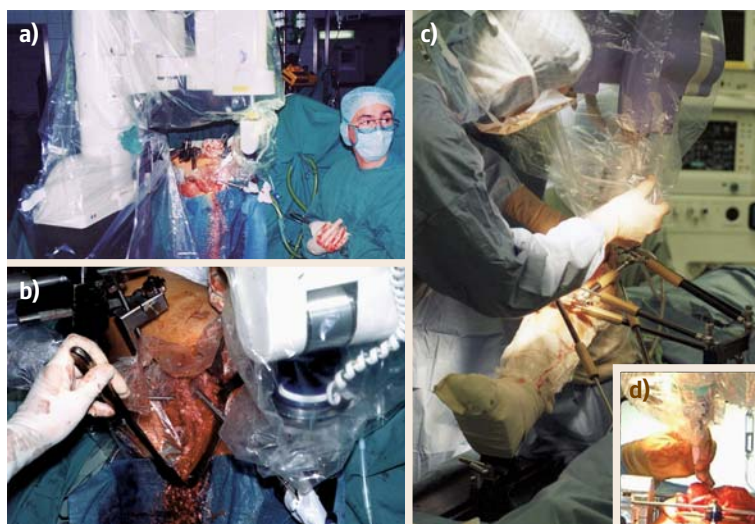


Fig. 52.5a–d Clinically deployed robots for orthopaedic surgery. **(a,b)** The Robodoc system [52.8, 9] represents the first clinically applied robot for joint reconstruction surgery and has been used for both primary and revision hip replacement surgery as well as knee replacement surgery. **(c,d)** The Acrobot system of Davies et al. [52.31] uses hands-on compliant guiding together with a form of virtual fixtures to prepare the femur and tibia for knee replacement surgery

surgical robots [52.16] showed that surgeons found this form of control to be very convenient and natural for surgical tasks. Subsequently, a number of groups have exploited this idea for precise surgical tasks, notably the *JHU Steady Hand* microsurgical robot [52.3] shown in Fig. 52.3 and the Imperial College Acrobot orthopaedic system [52.31] shown in Figs. 52.5c,d.

These control modes are not mutually exclusive and are frequently mixed. For example, the Robodoc system [52.8, 9] uses hands-on control to position the robot close to the patient's femur or knee and pre-programmed motions for bone machining. Similarly, the IBM/JHU LARS robot. [52.16] used both cooperative and telerobotic control modes. The cooperatively controlled Acrobot [52.31] uses preprogrammed virtual fixtures Sect. 52.1.3 derived from the implant shape and its planned position relative to medical images.

Each mode has advantages and limitations, depending on the task. Preprogrammed motions permit complex paths to be generated from relatively simple specifications of the specific task to be performed. They are most often encountered in surgical CAD/CAM applications where the planning uses two- (2-D) or three-dimensional (3-D) medical images. However, they can also provide useful *macro* motions combining sensory feedback in teleoperated or hands-on systems. Examples might include passing a suture or inserting a needle into a vessel after the surgeon has prepositioned the tip. On the other hand, interactive specification of motions based on real-

time visual appreciation of deforming anatomy would be very difficult.

Teleoperated control provides the greatest versatility for interactive surgery applications, such as dexterous MIS [52.2, 12, 17, 32] or remote surgery [52.33, 34]. It permits motions to be scaled, and (in some research systems) facilitates haptic feedback between master and slave systems. The main drawbacks are complexity, cost, and disruption to standard operating room work flow associated with having separate master and slave robots.

Hands-on control combines the precision, strength, and tremor-free motion of robotic devices with some of the immediacy of freehand surgical manipulation. These systems tend to be less expensive than telesurgical systems, since there is less hardware, and they can be easier to introduce into existing surgical settings. They exploit a surgeon's natural eye–hand coordination in an intuitively appealing way, and they can be adapted to provide force scaling [52.3, 4]. Although direct motion scaling is not possible, the fact that the tool moves in the direction that the surgeon pulls it makes this limitation relatively unimportant when working with a surgical microscope. The biggest drawbacks are that hands-on control is inherently incompatible with any degree of remoteness between the surgeon and the surgical tool and that it is not practical to provide hands-on control of instruments with distal dexterity.

Teleoperation and hands-on control are both compatible with *shared control modes* in which the robot controller constrains or augments the motions specified by the surgeon, as discussed in Sect. 52.2.3.

52.2.3 Virtual Fixtures and Human–Machine Cooperative Systems

Although one goal of both teleoperation and hands-on control is often *transparency*, i.e., the ability to move an instrument with the freedom and dexterity one might expect with a handheld tool, the fact that a computer is actually controlling the robot's motion creates many more possibilities. The simplest is a safety barrier or *no-fly zone*, in which the robot's tool is constrained from entering certain portions of its workspace. More sophisticated versions include virtual springs, dampers, or complex kinematic constraints that help a surgeon align a tool, maintain a desired force, or maintain a desired anatomical relationship. The Acrobot system shown in Figs. 52.5c,d represents a successful clinical application of the concept, which has many names, of which *virtual fixtures* seems to be the most popular [52.35, 36]. A number of groups are exploring extensions of the concept to active cooperative control, in which the surgeon and robot share or trade off control of the robot during a surgical task or subtask. As the ability of computers to model and follow along surgical tasks improves, these modes will become more and more important in surgical assistant applications. Figure 52.6 illustrates the overall concept of human–machine cooperative systems in surgery, and Fig. 52.7 illustrates the use of registered anatomical models to generate constraint-based virtual fixtures. These approaches are equally valid whether the surgeon interacts with the system through classical teleoperation or through hands-on compliant control. See also Chap. 31.

Both teleoperation and hands-on control are likewise used in human–machine cooperative systems for rehabilitation and disability assistance systems. Constrained hands-on systems offer special importance for rehabilitation applications and for helping people with movement disorders. Similarly, teleoperation and intelligent task following and control are likely to be vital for further advances in assistive systems for people with severe physical disabilities. See Chap. 53 for a further discussion of human–machine cooperation in assistive systems.

52.2.4 Safety and Sterility

Medical robots are safety-critical systems, and safety should be considered from the very beginning of the design process [52.37, 38]. Although there is some difference in detail, government regulatory bodies re-

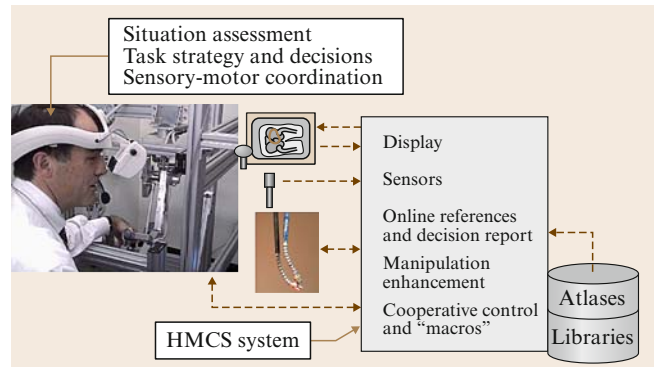


Fig. 52.6 Human–machine cooperative systems (HMCS) in surgery

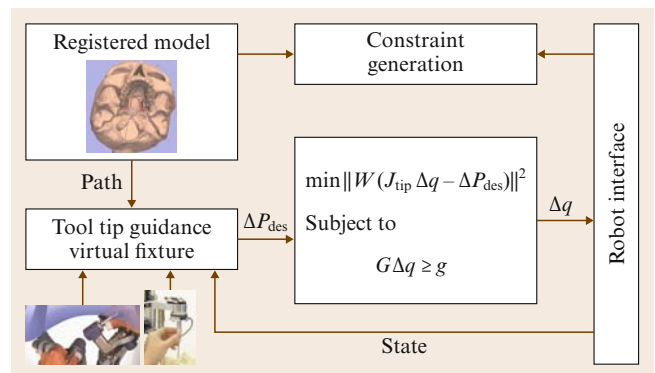


Fig. 52.7 Human–machine cooperative manipulation using constraint-based virtual fixtures, in which patient-specific constraints are derived from registered anatomical models [52.35]

quire a careful and rigorous development process with extensive documentation at all stages of design, implementation, testing, manufacturing, and field support. Generally, systems should have extensive redundancy built into hardware and control software, with multiple consistency conditions constantly enforced. The basic consideration is that no single point of failure should cause the robot to go out of control or to injure a patient. Although there is some difference of opinion as to the best way to make trade-offs, medical manipulators are usually equipped with redundant position encoders and ways to mechanically limit the speed and/or force that the robot can exert. If a consistency check failure is detected, two common approaches are to *freeze* robot motion or to cause the manipulator to go limp. Which is better depends strongly on the particular application.

Sterilizability and biocompatibility are also crucial considerations. Again, the details are application de-

pendent. Common sterilization methods include gamma rays (for disposable tools), autoclaving, soaking or gas sterilization, and the use of sterile drapes to cover unsterile components. Soaking or gas sterilization are less likely to damage robot components, but very rigorous cleaning is required to prevent extraneous foreign matter from shielding microbes from the sterilizing agent.

Careful attention to higher levels of application protocols is also essential. Just like any other tool, surgical robots must be used correctly by surgeons, and careful training is essential for safe practice. Surgeons must understand both the capabilities and limitations of the system. In surgical CAD/CAM applications, the surgeon must understand how the robot will execute the plan and be able to verify that the plan is being followed. If the surgeon is interactively commanding the robot, it is essential that the robot interpret these commands correctly. Similarly, it is essential that the robot's model of its task environment correspond correctly to the actual environment. Although careful design and implementation can practically eliminate the likelihood of a runaway condition by the manipulator, this will do little good if the robot is badly registered to the patient images used to control the procedure. If the robot fails for any reason, there must be well-documented and planned procedures for recovery (and possibly continuing the procedure manually).

Finally, it is important to remember that a well-designed robot system can actually *enhance* patient safety. The robot is not subject to fatigue or momentary lapses of attention. Its motions can be more precise and there is less chance that a slip of the scalpel may damage some delicate structure. In fact, the system can be programmed to provide *virtual fixtures* (Sect. 52.2.3) preventing a tool from entering a forbidden region unless the surgeon explicitly overrides the system.

52.2.5 Imaging and Modeling of Patients

As the capabilities of medical robots continue to evolve, the use of computer systems to model dynamically changing patient-specific anatomy will become increasingly important. There is a robust and diverse research community addressing a very broad range of research topics, including the creation of patient-specific models from medical images, techniques for updating these models based upon real-time image and other sensor data, and the use of these models for planning and monitoring of surgical procedures. Some of the pertinent research topics include the following:

- Medical image segmentation and image fusion to construct and update patient-specific anatomic models
- Biomechanical modeling for analyzing and predicting tissue deformations and functional factors affecting surgical planning, control, and rehabilitation
- Optimization methods for treatment planning and interactive control of systems
- Methods for registering the *virtual reality* of images and computational models to the *physical reality* of an actual patient
- Methods for characterizing treatment plans and individual task steps such as suturing, needle insertion, or limb manipulation for purposes of planning, monitoring, control, and intelligent assistance
- Real-time data fusion for such purposes as updating models from intraoperative images
- Methods for human-machine communication, including real-time visualization of data models, natural language understanding, gesture recognition, etc.
- Methods for characterizing uncertainties in data, models, and systems and for using this information in developing robust planning and control methods

An in-depth examination of this research is beyond the scope of this article. A more complete discussion of these topics may be found in the suggested further reading in Sect. 52.4.

52.2.6 Registration

Geometric relationships are fundamental in medical robotics, especially in surgical CAD/CAM. There is an extensive literature on techniques for coregistering coordinate systems associated with robots, sensors, images, and the patient [52.39, 40]. Following [52.40], we briefly summarize the main concepts here. Suppose that we have coordinates

$$\begin{aligned} \mathbf{v}_A^r &= (x_A, y_A, z_A) \\ \mathbf{v}_B^r &= (x_B, y_B, z_B), \end{aligned}$$

corresponding to comparable locations in two coordinate systems Ref_A and Ref_B . Then the process of registration is simply that of finding a function $T_{AB}(\cdot \cdot \cdot)$ such that

$$\mathbf{v}_B = T_{AB}(\mathbf{v}_A).$$

Generally, $T_{AB}(\dots)$ is assumed to be a rigid transformation of the form

$$T_{AB}(\mathbf{v}_A^r) = \mathbf{R}_{AB} \mathbf{v}_A^r + \mathbf{p}_{AB}^r,$$

where \mathbf{R}_{AB} represents a rotation and \mathbf{p}_{AB} represents a translation, but nonrigid transformations are becoming increasingly common. There are hundreds of methods for computing $T_{AB}(\dots)$. The most common for medical robotics involve finding a set of corresponding geometric features Γ_A and Γ_B whose coordinates can be determined in both coordinate systems and then finding a transformation that minimizes some distance function $d_{AB} = \text{distance}[\Gamma_B, T_{AB}(\Gamma_A)]$. Typical features can include artificial fiducial objects (pins, implanted spheres, rods, etc.) or anatomical features such as point landmarks, ridge curves, or surfaces.

One common class of methods is based on the iterated closest-point algorithm of *Besl* and *McKay* [52.41], for example, 3-D robot coordinates \mathbf{a}_j may be found for a collection of points known to be on the surface of an anatomical structure that can also be found in a segmented 3-D image. Given an estimate T_k of the transformation between image and robot coordinates, the method iteratively finds corresponding points \mathbf{b}_j on the surface that are closest to $T_k \mathbf{a}_j$ and then finds a new transformation

$$T_{k+1} = \arg \min_T \sum_j (\mathbf{b}_j - T \mathbf{a}_j)^2.$$

The process is repeated until some suitable termination condition is reached.

52.3 Systems, Research Areas, and Applications

52.3.1 Nonrobotic Computer-Assisted Surgery: Navigation and Image Overlay Devices

Medical robots are not ends in themselves. As the late Hap Paul often remarked, “the robot is a surgical tool designed to improve the efficacy of a procedure”. (Dr Paul was the founder of Integrated Surgical Systems. Along with William Bargar, he was one of the first people to recognize the potential of robots to fundamentally improve the precision of orthopaedic surgery.) In cases where the role of the robot is placing instruments on targets determined from medical images, *surgical navigation* is often a superior alternative. In surgical navigation [52.42], the positions of instruments relative to the reference markers on the patient are tracked using specialized electromechanical, optical, electromagnetic, or sonic digitizers or by more general computer vision techniques. After the relationships between key coordinate systems (patient anatomy, images, surgical tools, etc.) are determined through a registration process (Sect. 52.2.6), a computer workstation provides graphical feedback to the surgeon to assist in performing the planned task, usually by displaying instrument positions relative to medical images, as shown in Fig. 52.8a. Although the registration is usually performed computationally, a simple mechanical alignment of an image display with an imaging device can be surprisingly effective in some cases. One example [52.27] is shown in Fig. 52.8b.

The main advantages of surgical navigation systems are their versatility, their relative simplicity, and their ability to exploit the surgeon’s natural dexterity and haptic sensitivity. They are readily combined with *passive* fixtures and manipulation aids [52.43, 44]. The main drawbacks, compared to active robots, are those associated with human limitations in accuracy, strength, ability to work in certain imaging environments, and dexterity inside the patient’s body (Table 52.2).

Because these advantages often outweigh the limitations, surgical navigation systems are achieving widespread and increasing acceptance in such fields as neurosurgery, otolaryngology, and orthopaedics. Since much of the technology of these systems is compatible with surgical robots and since technical problems such as registration are common among all these systems, we may expect to see a growing number of hybrid applications combining medical robots and navigation.

52.3.2 Orthopaedic Systems

Orthopaedic surgery represents a natural surgical CAD/CAM application, and both surgical navigation systems and medical robots have been applied to orthopaedics. Bone is rigid and is easily imaged in CT and intraoperative X-rays, and surgeons are accustomed to doing at least some preplanning based on these images. Geometric accuracy in executing surgical plans is very important, for example, bones must be shaped accurately to ensure proper fit and positioning of components in

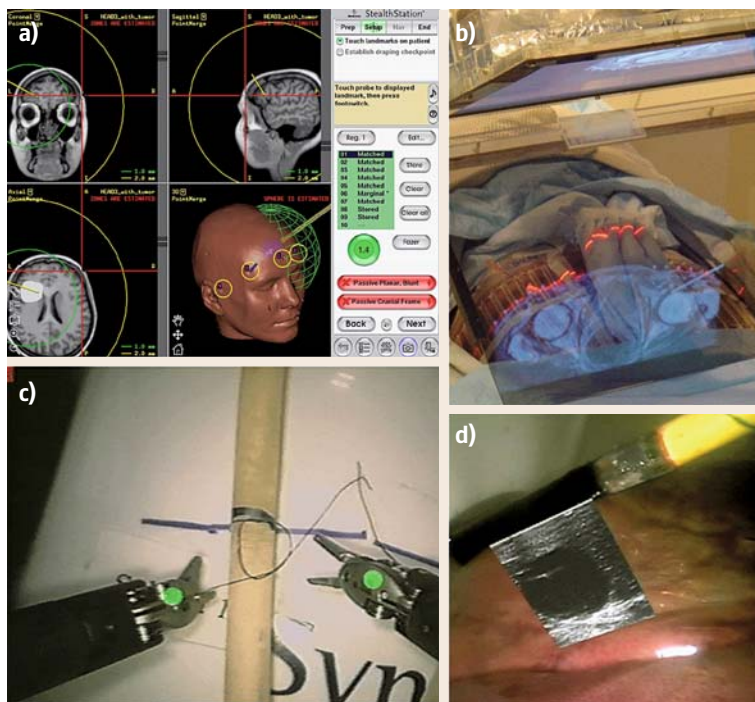


Fig. 52.8a–d Information enhancement for surgical assistance.

(a) Display from a typical surgical navigation system, here the Medtronic StealthStation; (b) the JHU image overlay system [52.45] uses a mirror to align the virtual image of a cross-sectional image with the corresponding physical position in the patient's body; (c) Sensory substitution display of surgical force information onto da Vinci surgical robot video monitor [52.46]; (d) Overlay of laparoscopic ultrasound onto the da Vinci surgical robot video monitor [52.47]

joint replacement surgery. Similarly, osteotomies require both accurate cutting and placement of bone fragments. Spine surgery often requires screws and other hardware to be placed into vertebrae without damage to the spinal cord, nerves, and nearby blood vessels.

The Robodoc system shown in Figs. 52.5a,b represents the first clinically applied robot for joint reconstruction surgery [52.8,9]. Since 1992, it has been applied successfully to both primary and revision hip replacement surgery, as well as knee surgery. Since this system exhibits many of the characteristics of surgical CAD/CAM, we will discuss it in some detail. In the *surgical CAD* phase, the surgeon selects the desired based on preoperative CT images and interactively specifies the desired position of the implant components. In the *surgical CAM* phase, surgery proceeds normally up to the point where the patient's bones are to be prepared to receive the implant. The robot is moved up to the operating table, the patient's bones are attached rigidly to the robot's base, and the robot is registered to the CT images either by use of implanted fiducial pins or by use of a 3-D digitizer to match bone surfaces to the CT images. After registration, the surgeon's hand guides the robot to an approximate initial starting position. Then, the robot autonomously machines the desired shape with a high-speed rotary cutter while the surgeon monitors progress.

During cutting, the robot monitors cutting forces, bone motion, and other safety sensor, and either the robot controller or the surgeon can pause execution at any time. If the procedure is paused for any reason, there are a number of error recovery procedures available to permit the procedure to be resumed or restarted at one of several defined checkpoints. Once the desired shape has been machined, surgery proceeds manually in the normal manner.

Subsequently, several other robotic systems for joint replacement surgery have been introduced or proposed. The references in Sect. 52.4 provide numerous examples. One notable system is the hands-on guided Acrobot system [52.31] for knee surgery shown in Figs. 52.5c,d. Similarly, several groups have recently proposed small orthopaedic robots attaching directly to the patient's bones [52.28].

52.3.3 Percutaneous Needle Placement Systems

Percutaneous (through the skin) needle placement has become ubiquitous in interventional radiology and surgery, as the least invasive actual surgical action practised today. These procedures fit naturally within the broader paradigm of surgical CAD/CAM systems. The

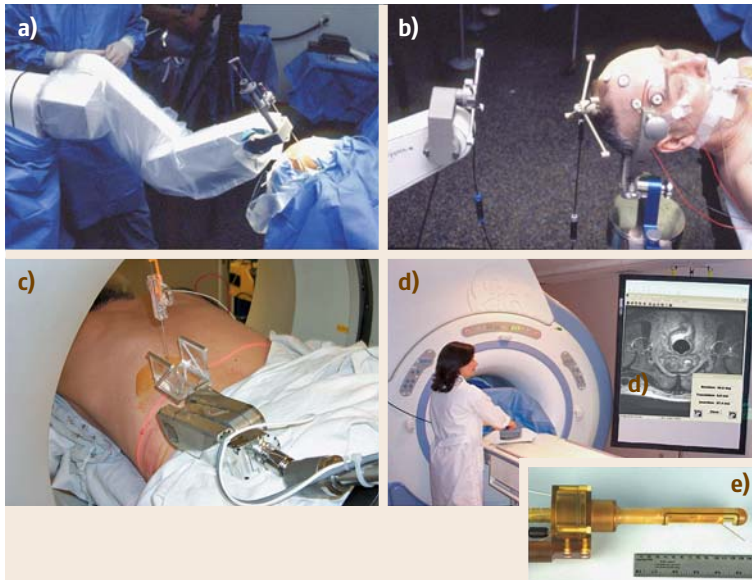


Fig. 52.9a–e Clinically deployed systems for in-scanner needle placement. (a,b) The Neuromate system [52.10] for stereotactic procedures in the brain uses a novel noncontact sensing system for robot-to-image registration; (c) Johns Hopkins system for in-CT needle placement [52.48, 49]; (d,e) Manually activated device for in-MRI transrectal needle placement into the prostate [52.50]

basic process involves use of patient images to identify targets within the patient and planning needle trajectories; inserting the needles and verifying their placement; performing some action such as an injection or taking a biopsy sample; and assessing the results. In most needle placements, an accuracy of 1–2 mm is acceptable, which is not easy to achieve freehand, because the target is not directly visible, soft tissues tend to deform and deflect, and needles often bend. Hence these procedures typically rely on some form of intraoperative imaging (X-ray fluoroscopy, CT, MRI, and ultrasound) for both guidance and verification. The surgical motion sequence involved in needle puncture typically has three phases:

1. touch down with the needle tip at the entry point
2. orientation of the needle by pivoting around this point
3. final insertion of the needle into the body along a straight trajectory

These motions are most often performed freehand, with varying degrees of information feedback for the physician. However, passive, semiautonomous, and active robotic systems are becoming more prevalent. Figure 52.9 shows several clinically deployed systems for needle placement.

Freehand Needle Placement Systems

Conventional freehand needle placement with CT and MRI guidance typically uses skin markers to locate the exact entry point [52.45], reference to the scanner's

alignment laser to control needle direction, and markers on the needle to control depth. With fluoroscopy, the typical procedure is to align the X-ray system so that it looks directly along the desired needle path, place the needle and stand it up so that its X-ray image is a dot, turn the X-ray system to look from the side, and insert the needle until it reaches the target. With ultrasound, the primary reliance is on surgeon experience or the use of some sort of needle guide to drive the needle to target while it passes in the ultrasound plane. In fact, a variety of simple mechanical guides have been tried out for use with all common image modalities. Several computer-assisted approaches have been proposed to enhance a human's ability to aim the needle during freehand needle placement. These include computer-aided laser guidance devices [52.51], augmented reality systems [52.52], and surgical navigation [52.42]. In addition, very simple variations of augmented reality have been developed for use with 2-D images such as CT or MRI slices [52.45]. B-mode ultrasound images (Fig. 52.8b) [52.53], where a semitransparent mirror is used together with a flat-panel display to ensure that the virtual image of the display is aligned to the corresponding physical section through the patient's anatomy, have also been used.

Passive and Semiautonomous Devices for Needle Placement

A few researchers [52.54] have proposed use of passive, encoded manipulator arms for image-guided needle

placement. After a registration step, these systems track the position and orientation of a passive needle guide and display the corresponding needle path in real time on **CT** or **MRI** images. Semiautonomous systems allow remote, interactive image-guided placement of the biopsy tool. For example, *Krieger et al.* performed transrectal prostate needle placement procedures in a conventional **MRI** environment [52.50] with a three-degree-of-freedom (DOF) manipulator. The device is driven from outside the bore with torsion cables, while the needle driver is tracked in the **MRI** system with active coils and monitored using an interactive graphical interface.

Active Robots for Needle Placement

Neurosurgery was one of the first clinical applications of active robots [52.10, 11, 13]. This is a natural application for surgical **CAD/CAM**. The entry and target points are planned on **CT/MRI** images, the robot coordinate system is registered to the image coordinate system (typically with fiducials affixed to the patient's head), and then the robot positions a needle or drill guide. The fiducial structure may be a conventional stereotactic head frame or, as in the Neuromate system [52.10], registration is achieved by simultaneous tracking of the robot and fiducials attached to the patient's skull. Special constraints of percutaneous access have led to the development of structures achieving *remote center of motion* (**RCM**) or fulcrum motion [52.15, 16]. In these systems, the **RCM** is positioned at the entry point, typically with an active Cartesian stage or a passive mechanical mechanism, and the robot sets the needle direction and (sometimes) the depth.

In order to speed up the sequence of imaging, planning, registration, and execution, one can exploit the robot's ability to work concurrently with imaging devices. This trend, again, began with intracranial applications. For extracranial use, *Stoianovici et al.* developed several variants of an **RCM**-based system deployed with X-ray fluoroscopy [52.15] and **CT** guidance [52.48, 49]. In [52.49], specialized fiducial structures have been incorporated in the needle driver to register the robot with a single image slice.

Magnetic resonance imaging (**MRI**) has an unmatched potential for guiding, monitoring, and controlling therapy, invoking intensive research on **MRI**-compatible robotic systems for needle placement [52.55] and for more complex interactive procedures [52.32].

Ultrasound (**US**) has many advantages for guiding needle placement and other interventional procedures. It is relatively inexpensive and compact, provides real-

time images, does not involve any ionizing radiation, and does not impose significant materials constraints on the robot design. Several robotic systems have been proposed for prostate interventions [52.56] using transrectal ultrasound guidance. There has also been some exploration of **US**-guided robotic systems for other (i. e., nonprostate) needle placement applications. Examples include experimental systems for liver [52.47, 57], gallbladder [52.58], and breast [52.59]. Figure 52.8d shows one example of the use of information overlay to assist in needle placement in a telesurgical application [52.47]. Whatever form of image feedback is available, steering flexible needles to hit desired targets while avoiding obstacles is a ubiquitous problem, having led to several novel approaches [52.60–62].

52.3.4 Telesurgical Systems

The concepts of telemedicine, telesurgery, and telepresence in surgery date from the 1970s. Since then, the potential for telesurgical systems to facilitate effective interventions in remote or hostile environments such as the battlefield, space, or thinly populated areas has continued to be recognized [52.63], and there have been some spectacular demonstrations including a transatlantic cholecystectomy [52.33] in 2001, as well as more nearly routine use in Canada [52.34].

However, the primary uses of telesurgical systems have been with the surgeon and patient in the same operating room. Teleoperated robots have been used for over 15 years in **MIS**, both as auxiliary surgical support systems to hold endoscopes or retractors [52.14, 16, 64, 65] and as *surgeon extender* systems to manipulate surgical instruments [52.2, 17]. There has also been recent work to develop telesurgical systems for use within imaging environments such as **MRI** [52.32].

A primary challenge for auxiliary support systems is to permit the surgeon to command the robot while his or her hands are otherwise occupied. Typical approaches have included conventional foot switches [52.14], instrument-mounted joysticks [52.16], voice control [52.7, 16], and computer vision [52.16, 66, 67].

A common goal in surgeon extender systems is to provide a measure of telepresence to the surgeon, specifically, to give the surgeon the sensation of performing open surgery from inside the patient. In early work, *Green et al.* [52.68] developed a successful prototype system for telesurgery combining remote manipulation, force feedback, stereoscopic imaging, ergonomic design, etc. Subsequently, several commercial telesurgical systems have been applied clinically for **MIS**. Of these,

Intuitive Surgical's daVinci [52.2] has been the most successful, with over 400 systems deployed as of 2007. Experience with these systems has demonstrated that a high-dexterity wrist is often critical for surgeon acceptance. Although originally targeted at cardiac surgery, as well as more general interventions, to date one of the most successful clinical applications has been in radical prostatectomies, where significant improvements in outcomes have been reported [52.69].

One emerging area for research exploits the inherent ability of telesurgical systems to act as *flight data recorders* during surgical procedures. Several authors [52.70–72] have begun analyzing such data for such purposes as measuring surgical skill, learning surgical gestures and motions, and providing data for surgical simulators.

52.3.5 Microsurgery Systems

Although *microsurgery* is not a consistently defined term, it generally indicates procedures performed on very small, delicate structures, such as those found in the eye, brain, spinal cord, small blood vessels, nerves, or the like. Microsurgical procedures are commonly performed under direct or video visualization, using some form of magnification (e.g., microscope, surgical loupes, high-magnification endoscope). The surgeon typically has little or no tactile appreciation of the forces being exerted by the surgical instruments and physiological hand tremor can be a significant factor limiting surgical performance.

There have been several efforts to compare micro-surgical anastomosis procedures using laparoscopic telesurgical systems to conventional microsurgery. *Schiff* et al. [52.73] among others reported significant reductions in tremor with either robot and significantly improved technical quality and operative times compared to conventional microsurgery. A number of groups have implemented telesurgery systems specifically for microsurgery [52.27, 74–76]. These systems are in various stages of development, from laboratory prototype to preliminary clinical experimentation.

Not all microsurgical robots are teleoperated. For example, the cooperatively controlled JHU Steady Hand robots [52.3, 4] shown in Fig. 52.3 are being developed for retinal, head-and-neck, neurosurgery, and other microsurgical applications. A modified version of this system has also been used for microinjections into single mouse embryos [52.77].

There have also been efforts to develop completely hand-held instruments that actively cancel physiolog-

ical tremor, for example, *Riviere* et al. [52.78] have developed an ophthalmic instrument using inertial sensors in the handle and adaptive filtering to estimate the tremulous component of instrument motion. A micromanipulator built into the instrument deflects the tip with an equal but opposite motion, compensating the tremor. Simple mechanical devices [52.79] for reducing tremor in specific tasks have also been developed.

An additional type of hand-held microsurgical and micro-therapeutic devices is reported in [52.80], which describes an active microendoscope for neuroendoscopy and therapy of the spinal cord able to safely navigate in the subarachnoid space and to avoid dangerous contact with the internal delicate structures thanks to a system based on hydrojets. Hydrojets come from the lateral surface of the catheter and, appropriately tuned and oriented, allow the tip of the endoscope to proceed without touching the spinal cord internal walls. The shared control system of the neuroendoscope, based on processing, segmentation, and analysis of the endoscopic images, assists the safe advancement of the tool in real time [52.81].

52.3.6 Endoluminal Robots

The term *endoluminal surgery* was first coined by *Cuschieri* et al. [52.82] as a major component of endoscopic surgery. Endoluminal procedures consist of bringing a set of advanced therapeutic and surgical tools to the area of interest by navigating in the lumina (i.e., the tube-like structures) of the human body, such as the gastrointestinal (GI) tract, the urinary tract, the circulatory system, etc. Currently, most endoluminal robots are designed for gastrointestinal applications, although there has been some initial work for other areas.

Traditionally, catheters and flexible endoscopes for endoluminal procedures have been inserted and manipulated manually from outside the body with the assistance of one or more visualization systems (e.g., direct endoscopic video, X-ray fluoroscopy, ultrasound). One major challenge is limited dexterity making it difficult to reach the desired target. Typically, flexible endoscopes have a bendable tip that can be steered by means of cable drives, and catheters may have only a fixed bend on a guide wire. There is also the inherent difficulty of *pushing a rope*. Once the target site is reached, these limitations become even more significant. Very simple instruments can be inserted through working channels or slid over guide wires, but dexterity is severely limited and there is no force feedback beyond what can be felt through the long, flexible instrument shaft.

These limitations have led a number of researchers to explore integration of more degrees of freedom in the catheter/endoscope body, as well as the design of intelligent tips with higher dexterity and sensing

capabilities. Early work by Ikuta et al. led to the development of a five-segment, 13 mm-diameter sigmoidoscope using shape-memory alloy (SMA) actuators. Subsequently, Ikuta developed 3 mm-diameter active endovascular devices using hydraulic actuators incorporating a novel band pass valve fabricated using micro-stereolithographic techniques [52.21].

Several examples exist of instrumented catheter tips with force sensors that allow the right branch of the circulatory systems to be found by estimating the force generated between the tip and the vessel walls. Basically, these sensorized endoluminal devices belong to the larger group of micro-electromechanical systems (MEMS)-instrumented surgical devices and the same sensing technologies can be also exploited for microsurgery. A recent survey article by *Rebello* [52.83] provides an excellent overview of sensorized catheters and other MEMS-based devices in endoluminal and microsurgical applications.

A third approach to endoluminal robots is represented by systems that move under their own power through the body, rather than being pushed. Early work on such systems is well summarized in [52.84]. In 1995 Burdick et al. developed an inchworm-like mechanism for use in the colon. This device combined a central extensor for propulsion and inflatable balloons for friction enhancement with the slippery colon tissue. A more advanced inchworm design for a semiautonomous robotic colonoscope was developed by *Dario* et al. [52.25] (Fig. 52.10). This device consists of a central silicone elongator, two clamping systems based on suction and gentle mechanical grasping of the colon tissue, and a silicone steerable tip integrating a complementary metal-oxide-semiconductor (CMOS) camera and a light-emitting diode (LED)-based illumination system. Thanks to its intrinsic flexibility, the robotic colonoscope applies forces on colon tissues that are ten times lower than those produced by traditional colonoscopes. More recent work on legged locomotion for GI applications Fig. 52.11c is reported in [52.20, 85]. Although the application is not endoluminal, the HeartLander system of *Riviere* et al. [52.23] shown in Figs. 52.11a,b shares many of the characteristics of these systems. It uses an inchworm-like gait to traverse the surface of the heart. Recently, Olympus has introduced a wireless gastrointestinal inspection system using external electromagnetic fields to manipulate a capsule camera [52.86]. An earlier application of electromagnetic manipulation of an object within the body was the *video tumor fighter* of *Ritter* et al. [52.87].

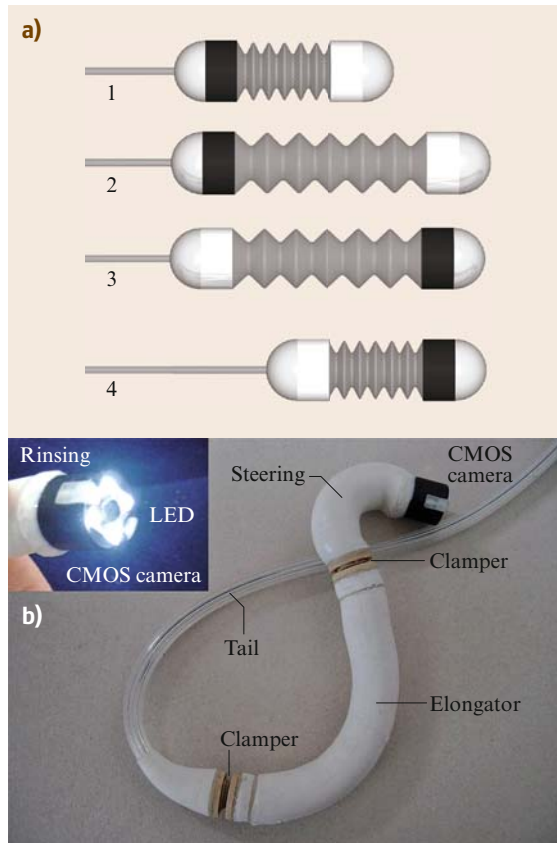


Fig. 52.10a,b Medical robot for colonoscopy [52.25]: (a) the gait cycle of the robot, consisting of: (1) proximal clamping, (2) elongation, (3) distal clamping, and (4) retraction; (b) a recent working prototype used for clinical trials

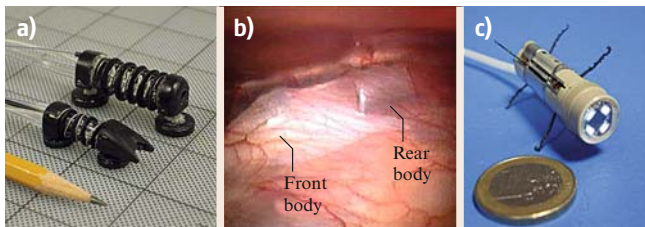


Fig. 52.11a-c Mobility inside the body. (a,b) HeartLander device for crawling across the surface of the heart [52.23]. (c) Legged capsule for gastrointestinal diagnosis and therapy [52.20]

52.3.7 Sensorized Instruments and Haptic Feedback

Surgical procedures almost always involve some form of physical interaction between surgical instruments and patient anatomy, and surgeons are accustomed to use their haptic appreciation of tool-to-tissue interaction forces in performing surgical procedures. In situations such as where the clumsiness of instrumentation or human sensory-motor limitations limit the surgeon's ability to feel these forces, surgeons have been trained to rely on visual or other cues to compensate. Apart from the need for haptic feedback for diagnostic procedures and tissue identification, it has been demonstrated that reduced tactile and force information can result in the application of unnecessarily large forces on the patient during surgery, with possible harm to the patient [52.88]. The quality of haptic feedback available in currently deployed surgical robots is poor or nonexistent. Current research addresses the limitations firstly by integrating force and other sensors into surgical end-effectors and secondly by developing improved methods for processing and conveying the sensed information to the surgeon.

Although the most obvious way to *display* haptic information in a telesurgical system is directly through the master hand controllers, this method has several limitations, including friction and limited bandwidth in the hand controllers. Although these issues may be addressed through specialized design (which may raise costs and require other compromises) and improved control, there has been considerable interest in *sensory substitution* schemes [52.89–91] in which force or other sensor information is displayed visually, aurally, or by other means. Figure 52.8c shows one example of sensory substitution for warning when a daVinci robot is about to break a suture [52.46].

Starting in the 1990s, several groups [52.91–93] have *sensorized* surgical instruments for microsurgery and MIS by equipping them with force sensors. Generally, these efforts relied on sensory substitution to display force data, either for freehand or telesurgical application. For example, *Poulose et al.* [52.92] demonstrated that a force sensing instrument used together with an IBM/JHU LARS robot [52.16] could significantly reduce both average retraction force and variability of retraction force during Nissen fundoplication. *Rosen et al.* [52.93] developed a force-controlled teleoperated endoscopic grasper equipped with position sensors and actuated by direct-current (DC) motors whose output torque is sensed and fed back through motors inte-

grated in a grasping handle. A similar approach was used by *Menciassi et al.* [52.94] for a microsurgery gripper equipped with semiconductor strain gauges and a PHANTOM (SensAble Technologies, Inc.) haptic interface.

Several researchers [52.95] have focused on specialized *fingers* and display devices for palpation tasks requiring delicate tactile feedback (e.g., for detecting hidden blood vessels or cancerous tissues beneath normal tissues). There has also been work to integrate nonhaptic sensors into surgical instruments, for example, *Fischer et al.* have developed instruments measuring both force and tissue oxygenation levels [52.96]. This information can be used for such purposes as assessing tissue viability, distinguishing between different tissue types, and controlling retraction so as not to cause ischemic tissue damage.

Finally, it is important to note that sensorized surgical tools have important application beyond their direct use in surgical procedures, for example, one use is in biomechanical studies to measure organ and tissue mechanical properties to improve surgical simulators.

52.3.8 Surgical Simulators and Telerobotic Systems for Training

Medical education is undergoing significant changes. The traditional paradigm for surgical technical training is often summarized as “see one, do one, teach one”. This method can be effective in open surgery, where the surgical trainee directly observes the expert surgeon hands, sees the instrument motion, and follows the organ manipulation. However, in endoscopic surgery it is difficult to observe the surgeon's hand movements (outside the body) and the surgical tool actions (inside the body and only visible on a video monitor). In addition, endoscopic surgery requires different skills than open surgery, such as spatial orientation, interpretation of 2-D images in 3-D, and manipulating instruments through entry portals. These considerations led to introduction of surgical simulation systems of varying degrees of complexity and realism for endoscopic and other minimally invasive procedures. Nowadays, training simulators have achieved widespread acceptance in the field of anaesthesia, intensive care, flexible endoscopy, surgery, interventional radiology, and other fields. The use of simulators for training is so common that working groups have been set up in order to evaluate these training systems based on shared guidelines [52.97] and many teaching hospitals have extensive simulation training centers.

A recent survey [52.98] divides training simulators into three groups, depending on the type of technology used: mechanical, hybrid, and virtual reality.

Mechanical simulators are boxes where objects or organs are placed and manipulated using surgical instruments. The manipulation is observed through a laparoscope and a screen. The simulated surgical task is observed by an experienced surgeon, who gives feedback to the trainee. Generally, there are no registration processes and the simulator must be reassembled after any training session. The *LapTrainer with SimuVision* (Simulab Inc., Seattle, USA) is a training system with a simulated laparoscope that consists of a boom-mounted digital camera in an open box trainer. The Johns Hopkins Urobotics has also developed mechanical simulators and a set of experiments for training in laparoscopy, which is also able to provide a quantifiable scale of dexterity.

A hybrid simulator uses a box with objects or organs as a mechanical simulator, but in addition the performance of the trainee are monitored by a computer which gives guidance for the task execution and an objective feedback based on preplanned metrics. Thanks to this feedback, the assistance and judgement of an experienced surgeon are not strictly necessary, for example, the ProMIS (Haptica Inc., Boston, USA) is a typical hybrid simulator for training basic skills such as suturing and knot tying. Surgical instruments are passed through dedicated ports and the trainee receives the same haptic feedback as in real surgery during manipulation in the box. In addition, ProMIS analyzes the trainee's performance by tracking the instrument position in 3-D and by measuring the execution time, path length, and smoothness of task execution.

Finally, *virtual-reality* training systems combine visualization and haptic interfaces to enable surgeons to interact efficiently and naturally with real-time computational models of patient anatomy [52.99]. The development of these systems is inherently a multidisciplinary effort, including real-time computer graphics, the development of high-bandwidth haptic devices, real-time biomechanical modeling of organs, tool-tissue interactions, expertise in training assessment, and human-machine information exchange, etc. [52.100]. Research in these areas is closely related to and synergistic with comparable developments in technology and systems for performing real interventions, for example, modeling of organ motion in response to forces is necessary to improve the accuracy of needle placement procedures. Haptic feedback devices must meet similar requirements whether the forces displayed are simulated

or measured directly in telesurgery, and so on. Finally, as noted earlier, *sensorized* instruments and real-time imaging are critical sources of data needed to create realistic biomechanical models.

The variety of interface devices and virtual reality laparoscopic simulators is quite wide and increasing numbers of systems are becoming commercially available. The Phantom interface is used in conjunction with virtual simulators to provide users with realistic haptic feedback (SensAble Technologies Inc., Woburn, MA, USA). The Xitact LS500 laparoscopy simulator (Xitact S.A., Lausanne, Switzerland) is a modular virtual-reality simulation platform with software for training and assessing performance in laparoscopic tasks. It is an open system including all or some of these subsystems: laparoscopic tools, mechanical abdomen, a personal computer (PC) providing the virtual-reality scenario, a haptic interface, a force feedback system and a tracking system for the tools. Several other systems for virtual reality simulation exist that exploit the hardware from Xitact or Immersion Medical, Inc. (Gaithersburg, MD, USA) and that are dedicated to specific surgical tasks: Lapmentor [52.101], the Surgical Education platform [52.102], LapSim [52.103], ProCedicus MIST [52.104], EndoTower [52.105], the Reachin laparoscopic trainer [52.106], Simendo [52.107], and the Vest system [52.108]. Specifically for training eye surgeons, the surgical simulator EYESi [52.109] uses advanced computer technology and virtual reality to simulate the feel of real eye surgery, making it possible for surgeons at all levels to acquire new skills and perfect their techniques in preparation for surgery on the human eye.

Making a comparison between these different categories of simulators is not trivial. Basically, box trainers and hybrid simulators require some experienced technicians for the set up and some organizational logistics due to legal and ethical factors related to the storage of freshly explanted organs. The most evident advantage of these simulators is that the tactile response from the manipulated objects is the same as in real surgery and complicated models of organs and tissue-tool interaction are not required. On the other hand, completely virtual-reality trainers are potentially very flexible, but they are limited by currently available computer and graphics hardware capabilities, as well as by the quality of current-generation anatomical and biomechanical models. Although demonstrations exist of the ability of simulators to record, objectively score, and hone the psychomotor skills of novice surgeons [52.110], the debate about the real improvement of surgical abilities is still open and conclusions are sometimes controversial.

52.3.9 Other Applications and Research Areas

The research areas described above illustrate major themes within medical robotics, but they are by no means

exhaustive. Many important application areas such as otolaryngology [52.111–114] and radiation therapy have necessarily been excluded for space reasons. For a fuller exploration, readers should consult the further reading suggestions in Sect. 52.4.

52.4 Conclusion and Future Directions

Medical robotics (and the larger field of *computer-integrated interventional medicine*) has great potential to revolutionize clinical practice by:

- exploiting technology to transcend human limitations in treating patients
- improving the safety, consistency, and overall quality of interventions
- improving the efficiency and cost-effectiveness of patient care
- improving training through the use of simulators, quantitative data capture and skill assessment methods, and the capture and playback of clinical cases
- promoting more effective use of information at all levels, both in treating individual patients and in improving treatment processes

From being the stuff of late-night comedy and science fiction 20 years ago, the field has reached a critical threshold, with clinically useful systems and commercial successes. The scope and number of research programs has grown exponentially in the past 5 years, and this chapter is by no means a comprehensive survey of the field. Interested readers are urged to refer to the further reading section for more complete treatments. In particular, the survey articles listed at the end of this section collectively contain somewhat fuller bibliographies than space permits here.

In the future, we can expect to see continued research in all aspects of technology and system development, with increasing emphasis on clinical applications. As this work proceeds, it is important that researchers remember several basic principles. The first, and arguably most important, principle is that medical robotics is fundamentally a *team* activity, involving academic researchers, clinicians, and industry. Each of these groups has unique expertise, and success comes from effective, highly interactive partnerships drawing upon this expertise. Building these teams takes a long-term commitment, and the successes in recent years are largely the pay-off from investments in creating these teams.

Second, it is important to work on problems with well-defined clinical and technical goals, in which the criteria for measuring success are ultimately related to real advantages in treating patients. In working toward these goals, it is important to have measurable and meaningful milestones and to emphasize rapid iteration with clinician involvement at all stages. Finally, it is essential that all team members recognize the level of commitment that is required to achieve success and that they enjoy what they are doing.

Books

- R. H. Taylor, S. Lavalée, G. Burdea, R. Mosges: *Computer-Integrated Surgery*, (MIT, Cambridge 1996)
- J. B. Stiehl, W. H. Konerman, R. G. Haaker: *Navigational and Robotics in Total Joint and Spine Surgery*, (Springer, Berlin Heidelberg 2003)
- A. DiGioia, B. Jaramaz, F. Picard, L. P. Nolte: *Computer and Robotic Assisted Knee and Hip Surgery*, (Oxford Univ. Press, Oxford 2004)
- R. Faust: *Surgical Robotics – History, Present and Future Applications* (Nova Sci., New York 2006)
- R. J. Maciunas: *Interactive Image-Guided Neurosurgery*, (Am. Assoc. Neurol. Surgeons, Rolling Meadows 1993)
- J. L. Prince and J. M. Links: *Medical Imaging Signals and Systems* (Prentice Hall, Upper Saddle River 2006)

Special Issues and Conference Proceedings

- *IEEE Trans. Robot. Autom.* Special Issue on Medical Robotics (2003)
- *IEEE Proceedings* Special Issue on Medical Robotics (2006)
- The *Proceedings of the IEEE Conference on Robotics and Automation* (IEEE Press) contain many medical robotics papers, especially in the years since 2003
- The *Proceedings of the International Symposia on Medical Image Computing and Computer Assisted*

Interventions, Lecture Notes in Computer Science (Springer, Berlin, Heidelberg) contain many medical robotics papers with special emphasis on image-guided systems

- The *Proceedings of the IEEE/RAS-EMBS International Conference Biomedical Robotics and Biomechatronics (BioRob)* report a new conference series, first held in Pisa in February 2006, with many pertinent papers.
- *Minimally Invasive Therapy* **14**, 4–5 (2005) This entire issue is devoted to surgical simulators and surgical training

Survey Articles

- R.H. Taylor, D. Stoianovici: Medical robotics in computer-integrated surgery, *IEEE Trans. Robot. Autom.* **19**, 765–781 (2003)
- R.H. Taylor, L. Joskowicz: Computer-integrated surgery and medical robotics. In: *Standard Handbook of Biomedical Engineering and Design*, ed. M. Kutz: (McGraw-Hill, New York 2002) 29.3–29.45.
- P. Dario, B. Hannaford, and A. Menciassi: Smart surgical tools and augmenting devices, *IEEE Trans. Robot. Autom.* **19**, 782–792 (2003)
- T. Peters: Image-guidance for surgical procedures, *Phys. Med. Biol.* **51**, R505–R540 (2006)
- J.B. Maintz, M.A. Viergever: A survey of medical image registration, *Med. Image Anal.* **2**(1) 1–37 (1998)
- S. Lavallee: Registration for computer-integrated surgery: methodology, state of the art. In: *Computer-Integrated Surgery*, ed. R.H. Taylor, S. Lavallee,

G. Burdea, R. Mosges (MIT Press, Cambridge 1996) pp. 77–98

- P. Kazanzides: Robots for orthopaedic joint reconstruction, In: *Robotics in Surgery: History, Current and Future Applications*, ed. R. A. Faust (Nova Science, New York 2007)
- P.B. McBeth, D.F. Louw, P.R. Rizun, G.R. Sutherland: Robotics in neurosurgery, *Am. J. Surg.* **188** 68S–75S (2004)
- D. Burschka, J.J. Corso, M. Dewan, W. Lau, M. Li, H. Lin, P. Marayoung, N. Ramey, G.D. Hager, B. Hoffman, D. Larkin, C. Hasser: Navigating inner space: 3-D assistance for minimally invasive surgery, *Robot. Aut. Syst.* **52** 5–26 (2005)
- D. Camarillo, T.M. Krummel, J.K. Salisbury: Robotic technology in surgery: past, present and future, *The Am. J. Surg.* **188** (2004)
- I. Kassim, L. Phee, W.S. Ng, G. Feng, P. Dario, C.A. Mosse: Locomotion techniques for robotic colonoscopy, *IEEE Eng. Med. Biol. Mag.* **25**(3), 49–56 (2006)
- R.H. Taylor: *A Perspective on Medical Robotics*, *IEEE Proc.*, **94**, 1652–1664 (2006)
- R.H. Taylor, P. Kazanzides: Medical robotics and computer-integrated interventional medicine. In: *Biomedical Information Technology* ed. D. Feng (Elsevier, 2007)
- K.J. Rebello: Applications of MEMS in surgery, *Proc. IEEE* **992**, 43–55 (2004)
- A. Liu, F. Tendick, K. Cleary, and C. Kaufmann: A survey of surgical simulation: applications, technology, and education, *Teleop. Virt. Environ.* **12**, 599–614 (2003)

References

- 52.1 R.H.L. Taylor Joskowicz: Computer-integrated surgery and medical robotics. In: *Standard Handbook of Biomedical Engineering and Design*, ed. by M. Kutz (McGraw Hill, New York 2003) pp. 29.23–29.45
- 52.2 G.S. Guthart, J.K. Salisbury: The intuitive telesurgery system: overview and application, *IEEE Int. Conf. Robot. Autom.* (ICRA2000) (San Francisco 2000) pp. 618–621
- 52.3 R. Taylor, P. Jensen, L. Whitcomb, A. Barnes, R. Kumar, D. Stoianovici, P. Gupta, Z.X. Wang, E. deJuan, L. Kavoussi: Steady-hand robotic system for microsurgical augmentation, *Int. J. Robot. Res.* **18**, 1201–1210 (1999)
- 52.4 D. Rothbaum, J. Roy, G. Hager, R. Taylor, L. Whitcomb: Task performance in stapedotomy: comparison between surgeons of different experience levels, *Otolaryngol. Head Neck Surg.* **128**, 71–77 (2003)
- 52.5 J.R. Adler, M.J. Murphy, S.D. Chang, S.L. Hancock: Image guided robotic radiosurgery, *Neurosurgery* **44**(6), 1299–1306 (1999)
- 52.6 A. Amin, J. Grossman, W. P. Early experience with a computerized robotically controlled catheter system, *J. Int. Card. Electrophysiol.*, Vol.12 (2005) pp.199–202
- 52.7 L. Mettler, M. Ibrahim, W. Jonat: One year of experience working with the aid of a robotic assistant (the voice-controlled optic holder AESOP) in gynaecological endoscopic surgery, *Hum. Reprod.* **13**, 2748–2750 (1998)

- 52.8 R.H. Taylor, H.A. Paul, P. Kazandzides, B.D. Mittelstadt, W. Hanson, J.F. Zuhars, B. Williamson, B.L. Musits, E. Glassman, W.L. Bargar: An image-directed robotic system for precise orthopaedic surgery, *IEEE Trans. Robot. Autom.*, Vol.10 (1994) pp.261–275
- 52.9 P. Kazanzides, B.D. Mittelstadt, B.L. Musits, W.L. Bargar, J.F. Zuhars, B. Williamson, P.W. Cain, E.J. Carbone: An integrated system for cementless hip replacement, *IEEE Eng. Med. Biol.* **14**, 307–313 (1995)
- 52.10 Q. Li, L. Zamorano, A. Pandya, R. Perez, J. Gong, F. Diaz: The application accuracy of the NeuroMate robot – A quantitative comparison with frameless and frame-based surgical localization systems, *Comput. Assisted Surg.* **7**(2), 90–98 (2002)
- 52.11 P. Cinquin, J. Troccaz, J. Demongeot, S. Lavallee, G. Champeboux, L. Brunie, F. Leitner, P. Sautot, B. Mazier, A. Perez, M. Djaid, T. Fortin, M. Chenic, A. Chapel: IGOR: image guided operating robot, *Innovation Technonogie Biologie Med.* **13**, 374–394 (1992)
- 52.12 H. Reichenspurner, R. Demaino, M. Mack, D. Boehm, H. Gulbins, C. Detter, B. Meiser, R. Ellgass, B. Reichart: Use of the voice controlled and computer-assisted surgical system Zeus for endoscopic coronary artery surgery bypass grafting, *J. Thoracic Cardiovasc. Surg.* **118**, 11–16 (1999)
- 52.13 Y.S. Kwoh, J. Hou, E.A. Jonckheere, S. Hayati: A robot with improved absolute positioning accuracy for CT guided stereotactic brain surgery, *IEEE Trans. Biomed. Eng.* **35**, 153–161 (1988)
- 52.14 J.M. Sackier, Y. Wang: Robotically assisted laparoscopic surgery. From concept to development, *Surg. Endosc.* **8**, 63–66 (1994)
- 52.15 D. Stoianovici, L. Whitcomb, J. Anderson, R. Taylor, L. Kavoussi: A modular surgical robotic system for image-guided percutaneous procedures, *Med. Image Comput. Comput.-Assisted Interventions (MICCAI-98)* (Cambridge 1998) pp.404–410
- 52.16 R.H. Taylor, J. Funda, B. Eldridge, K. Gruben, D. LaRose, S. Gomory, M. Talamini, L.R. Kavoussi, J. Anderson: A telerobotic assistant for laparoscopic surgery, *IEEE Eng. Med. Biol. Mag.* **14**, 279–287 (1995)
- 52.17 M. Mitsuishi, T. Watanabe, H. Nakanishi, T. Hori, H. Watanabe, B. Kramer: A telemicrosurgery system with colocated view and operation points and rotational-force-feedback-free master manipulator, 2nd Int. Symp. Med. Robot. Comput. Assisted Surg. (Baltimore 1995) pp.111–118
- 52.18 N. Simaan, R. Taylor, P. Flint: High dexterity snake-like robotic slaves for minimally invasive telesurgery of the throat, *Int. Symp. Med. Image Comput. Comput.-Assisted Interventions* (2004) pp.17–24
- 52.19 K. Ikuta, M. Tsukamoto, S. Hirose: Shape memory alloy servo actuator system with electric resistance feedback and application for active endoscope. In: *Computer-Integrated Surgery*, ed. by R.H. Taylor, S. Lavallee, G. Burdea, R. Mosges (MIT Press, Cambridge 1996) pp.277–282
- 52.20 A. Menciassi, C. Stefanini, S. Gorini, G. Pernorio, P. Dario, B. Kim, J.O. Park: Legged locomotion in the gastrointestinal tract problem analysis and preliminary technological activity, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (2004) pp. 937–942
- 52.21 K. Ikuta, H. Ichikawa, K. Suzuki, D. Yajima: Multi-degree of freedom hydraulic pressure driven safety active catheter, *IEEE Int. Conf. Robot. Autom.* (Orlando 2006) pp.4161–4166
- 52.22 S. Guo, J. Sawamoto, Q. Pan: A novel type of micro-robot for biomedical application, *IEEE/RSJ Int. Conf. Intell. Robots Syst.* (2005) pp.1047–1052, .
- 52.23 N. Patronik, C. Riviere, S.E. Qarra, M.A. Zenati: The heartlander: a novel epicardial crawling robot for myocardial injections, 19th Int. Congress Comput. Assisted Radiology Surg. (2005) pp.735–739
- 52.24 P. Dario, B. Hannaford, A. Menciassi: Smart surgical tools and augmenting devices, *IEEE Trans. Robot. Autom.*, Vol.19 (2003) pp.782–792
- 52.25 L. Phee, D. Accoto, A. Menciassi, C. Stefanini, M.C. Carrozza, P. Dario: Analysis and development of locomotion devices for the gastrointestinal tract, *IEEE Trans. Biomed. Eng.*, Vol. 49 (2002) pp.613–616
- 52.26 N. Suzuki, M. Hayashibe, A. Hattori: Development of a downsized master-slave surgical robot system for intragastric surgery, *ICRA Surg. Robot. Workshop* (Barcelona 2005)
- 52.27 K. Ikuta, K. Yamamoto, K. Sasaki: Development of remote microsurgery robot and new surgical procedure for deep and narrow space, *IEEE Conf. Robot. Autom.* (Taiwan 2003) pp.1103–1108
- 52.28 C. Plaskos, P. Cinquin, S. Lavallee, A.J. Hodgson: Praxiteles: a miniature bone-mounted robot for minimal access total knee arthroplasty, *Int. J. Med. Robot. Comput. Assisted Surg.*, Vol.1 (2005) pp.67–79
- 52.29 P.J. Berkelman, L. Whitcomb, R. Taylor, P. Jensen: A miniature microsurgical instrument tip force sensor for enhanced force feedback during robot-assisted manipulation, *IEEE Trans. Robot. Autom.*, Vol.19 (2003) pp.917–922
- 52.30 K. Chinzei, R. Gassert, E. Burdet: Workshop on MRI/fMRI compatible robot technology – a critical tool for neuroscience and image guided intervention, *IEEE Int. Conf. on Robot. Autom.* (Orlando 2006)
- 52.31 M. Jakopc, S.J. Harris, F.R.Y. Baena, P. Gomes, J. Cobb, B.L. Davies: The first clinical application of a hands-on robotic knee surgery system, *Comput. Aided Surg.* **6**, 329–339 (2001)
- 52.32 D.F. Louw, T. Fielding, P.B. McBeth, D. Gregoris, P. Newhook, G.R. Sutherland: Surgical robotics: A review and neurosurgical prototype development, *Neurosurgery* **54**, 525–537 (2004)

- 52.33 J. Marescaux, J. Leroy, M. Gagner, F. Rubino, D. Mutter, M. Vix, S.E. Butner, M.K. Smith: Transatlantic robot-assisted telesurgery, *Nature* **413**, 379–380 (2001)
- 52.34 M. Anvari, T. Broderick, H. Stein, T. Chapman, M. Ghodoussi, D.W. Birch, C. Mckinley, P. Trudeau, S. Dutta, C.H. Goldsmith: The impact of latency on surgical precision and task completion during robotic-assisted remote telepresence surgery, *Comput. Aided Surg.* **10**, 93–99 (2005)
- 52.35 M. Li, M. Ishii, R.H. Taylor: Spatial motion constraints in medical robot using virtual fixtures generated by anatomy, *IEEE Trans. Robot.* **23**, 4–19 (2007)
- 52.36 S. Park, R.D. Howe, D.F. Torchiana: Virtual fixtures for robotic cardiac surgery, *Fourth Int. Conf. Med. Image Comput. Computer-Assisted Intervention* (2001)
- 52.37 B. Davies: A discussion of safety issues for medical robots. In: *Computer-Integrated Surgery*, ed. by R. Taylor, S. Lavalée, G. Burdea, R. Moesges (MIT Press, Cambridge 1996) pp. 287–296
- 52.38 P. Varley: Techniques of development of safety-related software in surgical robots, *IEEE Trans. Inform. Technol. Biomed.* **3**, 261–267 (1999)
- 52.39 J.B. Maintz, M.A. Viergever: A survey of medical image registration, *Med. Image Anal.* **2**, 1–37 (1998)
- 52.40 S. Lavalée: Registration for computer-integrated surgery: methodology, state of the Art. In: *Computer-Integrated Surgery*, ed. by R.H. Taylor, S. Lavalée, G. Burdea, R. Mosges (MIT Press, Cambridge 1996) pp. 77–98
- 52.41 P.J. Besl, N.D. McKay: A method for registration of 3-D shapes, *IEEE Trans. Pattern Anal. Machine Intell.* **14**, 239–256 (1992)
- 52.42 R.J. Maciunas: *Interactive Image-Guided Neurosurgery* (Am. Association Neurological Surg., USA 1993)
- 52.43 R.H. Taylor, H.A. Paul, C.B. Cutting, B. Mittelstadt, W. Hanson, P. Kazanzides, B. Musits, Y.Y. Kim, A. Kalvin, B. Haddad, D. Khoramabadi, D. Larose: Augmentation of human precision in computer-integrated surgery, *Innov. Technol. Biol. Med.* **13**, 450–459 (1992)
- 52.44 J. Troccaz, M. Peshkin, B. Davies: The use of localizers, robots and synergistic devices in CAS, *CVRMed-MRCAS* **1205**, 727–736 (1997)
- 52.45 G. Fichtinger, A. Degeut, M. K, E. Balogh, G. Fischer, H. Mathieu, R.H. Taylor, S. Zinreich, L.M. Fayad: Image overlay guidance for needle insertion on CT scanner, *IEEE Trans. Biomed. Eng.* **52**, 1415–1424 (2005)
- 52.46 T. Akinbiyi, C.E. Reiley, S. Saha, D. Burschka, C.J. Hasser, D.D. Yuh, A.M. Okamura: Dynamic augmented reality for sensory substitution in robot-assisted surgical systems, *28th Annu. Int. Conf. IEEE Eng. Med. Biology Society* (2006) pp. 567–570
- 52.47 J. Leven, D. Burschka, R. Kumar, G. Zhang, S. Blumenkranz, X. Dai, M. Awad, G. Hager, M. Marohn, M. Choti, C. Hasser, R.H. Taylor: DaVinci canvas: a telerobotic surgical system with integrated, robot-assisted, laparoscopic ultrasound capability, *Med. Image Comput. Computer-Assisted Interventions* (Palm Springs 2005)
- 52.48 S. Solomon, A. Patriciu, R.H. Taylor, L. Kavoussi, D. Stoianovici: CT guided robotic needle biopsy: a precise sampling method minimizing radiation exposure, *Radiology* **225**, 277–282 (2002)
- 52.49 K. Masamune, G. Fichtinger, A. Patriciu, R. Susil, R. Taylor, L. Kavoussi, J. Anderson, I. Sakuma, T. Dohi, D. Stoianovici: System for robotically assisted percutaneous procedures with computed tomography guidance, *J. Comput. Assisted Surg.* **6**, 370–383 (2001)
- 52.50 A. Krieger, R.C. Susil, C. Menard, J.A. Coleman, G. Fichtinger, E. Atalar, L.L. Whitcomb: Design of a novel MRI compatible manipulator for image guided prostate intervention, *IEEE Trans. Biomed. Eng.* **52**, 306–313 (2005)
- 52.51 G.A. Krombach, T. Schmitz-Rode, B.B. Wein, J. Meyer, J.E. Wildberger, K. Brabant, R.W. Gunther: Potential of a new laser target system for percutaneous CT-guided nerve blocks: technical note, *Neuroradiology* **42**, 838–841 (2000)
- 52.52 M. Rosenthal, A. State, J. Lee, G. Hirota, J. Ackerman, K. Keller, E.D. Pisano, M. Jiroutek, K. Muller, H. Fuchs: Augmented reality guidance for needle biopsies: a randomized, controlled trial in phantoms, *Fourth Int. Conf. Med. Image Comput. Computer-Assisted Intervention* (2001) pp. 240–248
- 52.53 G. Stetten, V. Chib: Overlaying ultrasound images on direct vision, *J. Ultrasound Med.* **20**, 235–240 (2001)
- 52.54 H.F. Reinhardt: Neuronavigation: a ten years review. In: *Computer-Integrated Surgery*, ed. by R. Taylor, S. Lavalée, G. Burdea, R. Moesges (MIT Press, Cambridge 1996) pp. 329–342
- 52.55 E. Hempel, H. Fischer, L. Gumb, T. Hohn, H. Krause, U. Voges, H. Breitwieser, B. Gutmann, J. Durke, M. Bock, A. Melzer: An MRI-compatible surgical robot for precise radiological interventions, *Comput. Aided Surg.* **8**, 180–191 (2003)
- 52.56 Z. Wei, G. Wan, L. Gardi, G. Mills, D. Downey, A. Fenster: Robot-assisted 3D-TRUS guided prostate brachytherapy: system integration and validation, *Med. Phys.* **31**, 539–548 (2004)
- 52.57 E. Bector, G. Fischer, M. Choti, G. Fichtinger, R.H. Taylor: Dual-armed robotic system for intraoperative ultrasound guided hepatic ablative therapy: a prospective study, *IEEE Int. Conf. Robot. Autom.* (2004) pp. 377–382
- 52.58 J. Hong, T. Dohi, M. Hashizume, K. Konishi, N. Hata: An ultrasound-driven needle-insertion robot for percutaneous cholecystostomy, *Phys. Med. Biol.* **49**, 441–455 (2004)

- 52.59 G. Megali, O. Tonet, C. Stefanini, M. Boccardo, V. Papaspyropoulis, L. Angelini, P. Dario: A computer-assisted robotic ultrasound-guided biopsy system for video-assisted surgery, *Med. Image Comput. Computer-Assisted Intervention – MICCAI 2001 (Utrecht 2001)* pp. 343–350
- 52.60 S. Okazawa, R. Ebrahimi, J. Chuang, S. Salcudean, R. Rohling: Hand-held steerable needle device, *IEEE/ASME Trans. Mechatron.* **10**, 285–296 (2005)
- 52.61 R.J. Webster III, J.S. Kim, N.J. Cowan, G.S. Chirikjian, A.M. Okamura: Nonholonomic modeling of needle steering, *Int. J. Robot. Res.* **25**(5–6), 509–525 (2006)
- 52.62 J. Engh, G. Podnar, D. Kondziolka, C. Riviere: Toward effective needle steering in brain tissue, 28th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (2006) pp. 559–562
- 52.63 G.H. Ballantyne: Robotic surgery, telerobotic surgery, telepresence and telementoring, *Surg. Endosc.* **16**, 1389 (2002)
- 52.64 J.A. McEwen, C.R. Bussani, G.F. Auchinleck, M.J. Breault: Development and initial clinical evaluation of pre-robotic and robotic retraction systems for surgery, Second Workshop Med. Health Care Robot. (Newcastle 1989) pp. 91–101
- 52.65 P. Berkelman, P. Cinquin, J. Troccaz, J. Ayoubi, C. Letoublon, F. Bouchard: A compact, compliant laparoscopic endoscope manipulator, *IEEE Int. Conf. Robot. Aut.* (2002) pp. 1870–1875
- 52.66 A. Nishikawa, T. Hosoi, K. Koara, T. Dohi: FAcE MOUSE: A novel human-machine interface for controlling the position of a laparoscope, *IEEE Trans Robot. Autom.* **19**, 825–841 (2003)
- 52.67 A. Krupa, J. Gangloff, C. Doignon, M.F. deMathelin, G. Morel, J. Leroy, L. Soler, J. Marescaux: Autonomous 3-D positioning of surgical instruments in robotized laparoscopic surgery using visual servoing, *IEEE Trans. Robot. Autom.* **19**, 842–853 (2003)
- 52.68 P. Green, R. Satava, J. Hill, I. Simon: Telepresence: advanced teleoperator technology of minimally invasive surgery (abstract), *Surg. Endosc.* **6**, 90 (1992)
- 52.69 T. Ahlering, D. Woo, L. Eichel, D. Lee, R. Edwards, D. Skarecky: Robot-assisted versus open radical prostatectomy: a comparison of one surgeon's outcomes, *Urology* **63**, 819–822 (2004)
- 52.70 J. Rosen, J.D. Brown, L. Chang, M. Barreca, M. Sinanan, B. Hannaford: The BlueDRAGON – a system for measuring the kinematics and the dynamics of minimally invasive surgical tools in-vivo, *IEEE Int. Conf. Robot. Autom.* (2002) pp. 1876–1881
- 52.71 H.C. Lin, I. Shafraan, T.E. Murphy, A.M. Okamura, D.D. Yuh, G.D. Hager: Automatic detection and segmentation of robot-assisted surgical motions, *MICCAI* (2005) pp. 802–810
- 52.72 H. Mayer, F. Gomez, D. Wierstra, I. Nagy, A. Knoll, J. Schmidhuber: A system for robotic heart surgery that learns to tie the knots using recurrent neural networks, *Int. Conf. Intell. Robot. Syst. (IROS)* (2006) pp. 543–548
- 52.73 J. Schiff, P. Li, M. Goldstein: Robotic microsurgical vasovasostomy and vasoepididymostomy: a prospective randomized study in a rat model, *J. Urol.* **171**, 1720–1725 (2004)
- 52.74 P.S. Jensen, K.W. Grace, R. Attariwala, J.E. Colgate, M.R. Glucksberg: Toward robot assisted vascular microsurgery in the retina, *Graefes Arch. Clin. Exp. Ophthalmol.* **235**, 696–701 (1997)
- 52.75 H. Das, H. Zak, J. Johnson, J. Crouch, D. Frambaugh: Evaluation of a telerobotic system to assist surgeons in microsurgery, *Comput. Aided Surg.* **4**, 15–25 (1999)
- 52.76 K. Hongo, S. Kobayashi, Y. Kakizawa, J.I. Koyama, T. Goto, H. Okudera, K. Kan, M.G. Fujie, H. Iseki, K. Takakura: NeuRobot: telecontrolled micromanipulator system for minimally invasive microneurosurgery—preliminary results, *Neurosurgery* **51**, 985–988 (2002)
- 52.77 A. Kapoor, R. Kumar, R. Taylor: Simple biomanipulation tasks with a steady hand cooperative manipulator, Sixth Int. Conf. Med. Image Comput. Computer Assisted Intervention – MICCAI 2003 (Montreal 2003) pp. 141–148
- 52.78 C. Riviere, J. Gangloff, M. de Mathelin: Robotic compensation of biological motion to enhance surgical accuracy, *Proc IEEE* **94**(9), 1705–1716 (2006)
- 52.79 W. Armstrong, A. Karamzadeh, R. Crumley, T. Kelley, R. Jackson, B. Wong: A novel laryngoscope instrument stabilizer for operative microlaryngoscopy, *Otolaryngol. Head Neck Surg.* **132**, 471–477 (2004)
- 52.80 L. Ascari, C. Stefanini, A. Menciassi, S. Sahoo, P. Rabischong, P. Dario: A new active microendoscope for exploring the sub-arachnoid space in the spinal cord, *IEEE Conf. Robot. Autom.* (2003) pp. 2657–2662
- 52.81 U. Bertocchi, L. Ascari, C. Stefanini, C. Laschi, P. Dario: Human-robot shared control for robot-assisted endoscopy of the spinal cord, *IEEE/RAS-EMBS Int. Conf. Biomed. Robot. Biomechatron.* (2006) pp. 543–548
- 52.82 A. Cuschieri, G. Buess, J. Perissat: *Operative Manual of Endoscopic Surgery* (Springer, Heidelberg 1992)
- 52.83 K.J. Rebello: Applications of MEMS in surgery, *IE, Vol. 992* (2004) pp. 43–55
- 52.84 I. Kassim, W.S. Ng, G. Feng, S.J. Phee: Review of locomotion techniques for robotic colonoscopy, *Int. Conf. Robot. Autom.* (Taipei 2003) pp. 1086–1091
- 52.85 C. Stefanini, A. Menciassi, P. Dario: Modeling and experiments on a legged microrobot locomoting in a tubular, compliant and slippery environment, *Int. J. Robot. Res.* **25**, 551–560 (2006)
- 52.86 Y. Kusuda: A further step beyond wireless capsule endoscopy, *Sensor Rev.* **25**, 259–260 (2005)
- 52.87 R.C. Ritter, M.S. Grady, M.A. Howard, G.T. Gilles: Magnetic stereotaxis: computer assisted, image guided remote movement of implants in the brain. In: *Computer Integrated Surgery*, ed. by R.H. Tay-

- lor, S. Lavallee, G.C. Burdea, R. Mosges (MIT Press, Cambridge 1995) pp. 363–369
- 52.88 C.R. Wagner, N. Stylopoulos, P.G. Jackson, R.D. Howe: The benefit of force feedback in surgery: examination of blunt dissection, *Presence: Teleoperators Virtual Environments* (2007)
- 52.89 A.M. Okamura: Methods for haptic feedback in teleoperated robot-assisted surgery, *Ind. Robot* **31**, 499–508 (2004)
- 52.90 M.J. Massimino: Improved force perception through sensory substitution, *Contr. Eng. Practice* **3**, 215–222 (1995)
- 52.91 P. Gupta, P. Jensen, E. de Juan: Quantification of tactile sensation during retinal microsurgery, MICCAI99: Second Int. Conf. Med. Image Comput. Computer-Assisted Intervention (Cambridge 1999)
- 52.92 P.K. Poulouse, M.F. Kutka, M. Mendoza-Sagaon, A.C. Barnes, C. Yang, R.H. Taylor, M.A. Talamini: Human versus robotic organ retraction during laparoscopic nissen fundoplication, *Surg. Endosc.* **13**, 461–465 (1999)
- 52.93 J. Rosen, B. Hannaford, M. MacFarlane, M. Sinanan: Force controlled and teleoperated endoscopic grasper for minimally invasive surgery – experimental performance evaluation, *IEEE Trans. Biomed. Eng.* **46**, 1212–1221 (1999)
- 52.94 A. Menciassi, A. Eisnberg, M.C. Carrozza, P. Dario: Force sensing microinstrument for measuring tissue properties and pulse in microsurgery, *IEEE/ASME Trans. Mechatron.* **8**, 10–17 (2003)
- 52.95 R.D. Howe, W.J. Peine, D.A. Kontarinis, J.S. Son: Remote palpation technology, *IEEE Eng. Med. Biol.* **14**(3), 318–323 (1995)
- 52.96 G. Fischer, T. Akinbiyi, S. Saha, J. Zand, M. Talamini, M. Marohn, R. Taylor: Ischemia and force sensing surgical instruments for augmenting available surgeon information, *IEEE Int. Conf. Biomed. Robot. Biomechatron.* – BioRob 2006 (Pisa 2006)
- 52.97 F.J. Carter, M.P. Schijven, R. Aggarwal, T. Grantcharov, N.K. Francis, G.B. Hanna, J.J. Jakimowicz: Consensus guidelines for validation of virtual reality surgical simulators, *Surg. Endo.* **19**, 1523–1532 (2005)
- 52.98 F.H. Halvorsen, O.J. Elle, E. Fosse: Simulators in surgery, *Minimally Invasive Ther.* **14**, 214–223 (2005)
- 52.99 K. Moorthy, Y. Munz, S.K. Sarker, A. Darzi: Objective assessment of technical skills in surgery, *Br. Med. J.* **327**, 1032–1037 (2003)
- 52.100 C. Basdogan, S. De, J. Kim, M. Muniyandi, H. Kim, M.A. Srinivasan: Haptics in Minimally Invasive Surgical Simulation and Training, *IEEE Comput. Graphics Appl.* **24**(2), 56–64 (2004)
- 52.101 Lapmentor: www.simbionix.com
- 52.102 Surgical Education platform: www.meti.com
- 52.103 LapSim: www.surgical-science.com
- 52.104 MIST: www.mentice.com
- 52.105 EndoTower: www.verifi.com
- 52.106 Laparoscopic Trainer: www.reachin.se
- 52.107 Simendo: www.simendo.nl
- 52.108 Vest System: www.select-it.de
- 52.109 EYESI: <http://www.vrmagic.com/>
- 52.110 F. Cavallo, G. Megali, S. Sinigaglia, O. Tonet, P. Dario: A biomechanical analysis of surgeon's gesture in a laparoscopic virtual scenario, *Studies Health Techn. Informatics* **119**, 79–84 (2006)
- 52.111 K.T. Kavanagh: Applications of image-directed robotics in otolaryngologic surgery, *Laryngoscope* **104**, 283–293 (1994)
- 52.112 J. Wurm, H. Steinhart, K. Bumm, M. Vogelee, C. Nimsky, H. Iro: A novel robot system for fully automated paranasal sinus surgery, *CARS 2003. Comput. Assisted Radiol. Surg.* (2003) pp. 633–638
- 52.113 M. Li, M. Ishii, R.H. Taylor: Spatial motion constraints in medical robot using virtual fixtures generated by anatomy, *IEEE Trans. Robot.* **2**, 1270–1275 (2006)
- 52.114 G. Strauss, K. Koulechov, R. Richter, A. Dietz, C. Trantakis, T. Lueth: Navigated control in functional endoscopic sinus surgery, *Int. J. Med. Robot. Comput. Assisted Surg.* **1**, 31–41 (2005)

53. Rehabilitation and Health Care Robotics

H.F. Machiel Van der Loos, David J. Reinkensmeyer

The field of rehabilitation robotics develops robotic systems that assist persons who have a disability with necessary activities, or that provide therapy for persons seeking to improve physical or cognitive function. This chapter will discuss both of these domains and provide descriptions of the major achievements of the field over its short history. Specifically, after providing background information on world demographics (Sect. 53.1.2) and the history (Sect. 53.1.3) of the field, Sect. 53.2 describes physical therapy and training robots, and Sect. 53.3 describes robotic aids for people with disabilities. Section 53.4 then briefly discusses recent advances in smart prostheses and orthoses that are related to rehabilitation robotics. Finally, Sect. 53.5 provides an overview of recent work in diagnosis and monitoring for rehabilitation as well as other health-care issues. At the conclusion of this chapter, the reader will be familiar with the history of rehabilitation robotics and its primary accomplishments, and will understand the challenges the field faces in the future as it seeks to improve health care and the well-being of persons with disabilities. In this chapter, we will describe an application of robotics that may in the future touch many of us in an acutely personal way.

53.1 Overview	1223
53.1.1 Taxonomy of Rehabilitation Robotics	1224
53.1.2 World Demographics	1224
53.1.3 Short History of the Field of Rehabilitation Robotics	1225
53.2 Physical Therapy and Training Robots	1227
53.2.1 Grand Challenges and Roadblocks ..	1227
53.2.2 Movement Therapy after Neurologic Injury	1228
53.2.3 Robotic Therapy for the Upper Extremity	1229
53.2.4 Robotic Therapy for Walking	1231
53.3 Aids for People with Disabilities	1235
53.3.1 Grand Challenges and Enabling Technologies	1235
53.3.2 Types and Examples of Assistive Rehabilitation Robots ..	1236
53.4 Smart Prostheses and Orthoses	1240
53.4.1 Grand Challenges and Roadblocks ..	1240
53.4.2 Targeted Reinnervation	1240
53.4.3 Brain–Machine Interfaces	1241
53.4.4 Advances in Neural Stimulation	1241
53.4.5 Embedded Intelligence	1242
53.5 Augmentation for Diagnosis and Monitoring	1242
53.5.1 Introduction: Grand Challenges and Enabling Technologies	1242
53.5.2 Smart Clinics with Automated Health Care Monitoring and Care	1243
53.5.3 Home–Based Rehabilitation Monitoring Systems	1243
53.5.4 Wearable Monitoring Devices	1244
53.6 Safety, Ethics, Access, and Economics	1244
53.7 Conclusions and Further Readings	1245
References	1246

53.1 Overview

When we become unable to interact physically with our immediate environment as we desire in order to achieve our personal goals through injury or disease, or

when one of our family members, friends or neighbors is in this situation, we seek technology-based solutions to assist us in relearning how to complete our activi-

ties of daily living (ADLs), or to assist us in actually doing them if we are unable to relearn. While human therapists and attendants can indeed provide the types of assistance required, the projected short-term demographics of China, Japan, and the Scandinavian countries show a growing shortage of working-age adults. Age-related disabilities will soon dominate the service sector job market, put many older and disabled people at risk, and increase the need for institutionalization when there is no viable home-based solution. National programs to develop personal robots, robotic therapy, smart prostheses, smart beds, smart homes, and tele-rehabilitation services have accelerated in the past ten years and will need to continue apace with the ever-increasing ability of health care to allow people to live longer through the repression of disease and improvements in surgical and medication interventions. Rehabilitation robotics, although only a 40-year-old discipline [53.1–3], is projected to grow quickly in the coming decades.

53.1.1 Taxonomy of Rehabilitation Robotics

The field of rehabilitation robotics is generally divided into the categories of *therapy* and *assistance robots*. In addition, rehabilitation robotics includes aspects of artificial limb (prosthetics) development, functional neural stimulation, (FNS) and technology for the diagnosis and monitoring of people during ADLs.

Therapy robots generally have at least two main users simultaneously, the person with a disability who is receiving the therapy and the therapist who sets up and monitors the interaction with the robot. Types of therapy that have benefited from robotic assistance are upper- and lower-extremity movement therapy, enabling communication for children with autism, and enabling exploration (education) for children with cerebral palsy (CP) or other developmental disabilities. A robot may be a good alternative to a physical or occupational therapist for the actual hands-on intervention for several reasons: (1) once properly set up, an automated exercise machine can consistently apply therapy over long periods of time without tiring; (2) the robot's sensors can measure the work performed by the patient and quantify, to an extent perhaps not yet measurable by clinical scales, any recovery of function that may have occurred, which may be highly motivating for a person to continue with the therapy; and (3) the robot may be able to engage the patient in types of therapy exercises that a therapist cannot do, such as magnifying movement errors to provoke adaptation [53.4, 5].

Assistive robots are generally grouped according to whether they focus on manipulation, mobility, or cognition. Manipulation aids are further classified into fixed-platform, portable-platform, and mobile autonomous types. Fixed-platform robots perform functions in the kitchen, on the desktop, or by the bed. Portable types are manipulator arms attached to an electric wheelchair to hold and move objects and to interact with other devices and equipment, as in opening a door. Mobile autonomous robots can be controlled by voice or other means to carry out manipulation and other errands in the home or workplace. Mobility aids are divided into electric wheelchairs with navigation systems and mobile robots that act as smart, motorized walkers, allowing people with mobility impairments to lean on them to prevent falls and provide stability. The third main type, cognitive aids, assist people who have dementia, autism or other disorders that affect communication and physical well-being.

The fields of prosthetics and FNS are closely allied with rehabilitation robotics. Prostheses are artificial hands, arms, legs, and feet that are worn by the user to replace amputated limbs. Prostheses are increasingly incorporating robotic features. FNS systems seek to reanimate the limb movements of weak or paralyzed people by electrically stimulating nerve and muscle. FNS control systems are analogous to robotic control systems, except that the actuators being controlled are human muscles. Another related field is technology for monitoring and diagnosing health care issues as a person performs ADLs.

The chapter is organized according to this taxonomy. After providing background information on world demographics (Sect. 53.1.2) and the history (Sect. 53.1.3) of the field, Sect. 53.2 describes physical therapy and training robots, and Sect. 53.3 describes robotic aids for people with disabilities. Section 53.4 then reviews recent advances in smart prostheses and orthoses that are related to rehabilitation robotics. Finally, Sect. 53.5 provides an overview of recent work in diagnosis and monitoring for rehabilitation as well as other health care issues.

53.1.2 World Demographics

The various areas of rehabilitation robotics focus on different user populations, but a common characteristic of these populations is that they have a disability. Disability is defined in the *Americans with Disabilities Act* as “a physical or mental impairment that substantially limits one or more of the major life activities.”

Table 53.1 Prevalence and incidence of disability and aging in selected countries [53.6]

Country	Number of people with disabilities	Percentage of population with disabilities	Number of elderly people	Percentage of population that is elderly
France	5 146 000	8.3	12 151 000	19.6
USA	52 591 000	20.0	35 000 000	12.4
Great Britain	4 453 000	7.3	12 200 000	29.5
The Netherlands	1 432 000	9.5	2 118 808	13.4
Spain	3 528 220	8.9	6 936 000	17.6
Japan	5 136 000	4.3	44 982 000	35.7
Korea	3 195 000	7.1	16 300 000	36.0

In the industrialized countries (e.g., Japan, US, Canada, and Europe), the incidence of disability varies between 8% and 20%, with differences likely due primarily to varying definitions of disability and reporting conventions (Table 53.1). Age is a risk factor for disability, and lower birth rates and life-extending health care are the dominant factors contributing to the aging of the population and a concomitant rise in disability. In China, the population control policies of the 1970s have created a lack of working-age adults to support the economy. The disproportionate incidence of disability in the elderly population makes it clear that developers of rehabilitation robotics will also be faced with users who, as a demographic group, generally have lower levels of sensory and motor capability, and may have impaired cognition as well. The urgency of making advances in this field is increasing in line with these demographic changes.

53.1.3 Short History of the Field of Rehabilitation Robotics

The history of rehabilitation robotics is almost as old as that of robotics itself, although emanating from very different sources. Several books, chapters, and papers have been written on the history of rehabilitation robotics in more detail than this section [53.1, 7, 8], and numerous papers in the proceedings of the Institute of Electrical and Electronics Engineers (IEEE) International Conference on Rehabilitation Robotics also provide more grounding for historical perspective. The chronology below pays particular attention to early work and to projects with notable clinical and/or commercial impact.

Early robotics, starting in the late 1950s, focused on large manipulators to replace workers in factories for dirty, dangerous, and undesirable tasks. The earliest rehabilitation robots came from the field of prosthetics and orthotics (P&O). The Case Western University arm (1960s) and the Rancho Los Amigos *Golden Arm*

(early 1970s) (reviewed in [53.7]) were both adaptations of replacement mechanical arms meant as powered orthoses [53.1]. The user drove the *Golden Arm* with a set of tongue-operated switches, joint-by-joint, an arduous means of endpoint control. In the mid 1970s, the Department of Veterans Affairs began funding a group at the Applied Physics Lab under the guidance of Seamone and Schmeisser to computerize an orthosis mounted on a workstation to do activities of daily living (ADL) tasks such as feeding a person and turning pages [53.9]. For the first time, a rehabilitation robot had a command-type interface, not just a joint-by-joint motion controller.

The 1970s also saw the French *Spartacus* system being developed, guided by the vision of Jean Vertut, for use by people with high-level spinal cord injury as well as children with cerebral palsy [53.10]. This system did not emerge from the P&O field but was developed by the French Atomic Energy Commission (CEA), which used large telemanipulators for nuclear fuel rod handling. One of these was adapted so that people with movement impairment could control it using a joystick for pick-and-place tasks. A decade later, one of the researchers on the *Spartacus* project, Hok Kwee, began the *MANUS* project, a dedicated effort to develop the first wheelchair-mounted manipulator designed expressly as a rehabilitation robot, not adapted from a design from another field.

However, in between, several other major programs were begun. In 1978, at Stanford University, and then with decades-long funding from the US Department of Veterans Affairs, Larry Leifer started the vocational assistant robot program, culminating in several clinically tested versions of the desktop vocational assistant robot (*DeVAR*) [53.3, 11, 12], a mobile version, the mobile vocational assistant robot (*MoVAR*) [53.13], and finally the professional vocational assistant robot (*ProVAR*), which had the advanced ability for the user to program tasks in an easy-to-use browser-type environment [53.14]. This step was made since, although *DeVAR* made it briefly

onto the market in the early 1990s, multisite user testing revealed it was still too costly for the functionality it had: ProVAR development ensued, then continued by Machiel Van der Loos. All these versions were based on the Puma-260 industrial manipulator to achieve robust, safe operation. Research shifted in 2006 to the Veterans Affairs (VA) in Syracuse, NY, to integrate sensing and autonomous features and explore new, more cost-effective manipulator options.

In the mid 1980s, from observations on the unsuitability of existing industrial, educational, and orthosis-derived manipulators for rehabilitation applications, Tim Jones at Universal Machine Intelligence (later Oxford Intelligent Machines, OxIM) in the UK began an intensive effort to provide the rehabilitation robotics community with its first workhorse system specially designed from the ground up for human service tasks. Over ten years, a series of systems, starting with the RTX model, were used in numerous research labs and clinics around the world. The most extensive effort to use the OxIM arm was in France, and a suite of research projects, funded by the French government and the European Research Commission, starting as the robot for assisting the integration of the disabled (RAID), then as MASTER [53.15], developed and clinically tested workstation-based assistive systems based on the RTX and subsequent OxIM arms. When OxIM ceased building its arms, the French company Afma Robotics [53.16] took over efforts to commercialize the MASTER system, which it continues to do today (2007).

The UK was also the site of the first commercially available feeding robot, Handy-I, an inexpensive and well-received device first designed by Mike Topping and then commercialized by Rehabilitation Robotics, Ltd. in the 1990s [53.17]. Primarily aimed to enable people with cerebral palsy to achieve a measure of independence in feeding themselves, task environments later also included face washing and the application of cosmetics, areas of high demand identified by its users.

The history of mobile manipulator applications began in the 1980s with adaptations of educational and industrial robots, and achieved a boost with the funding of the US National Institute on Disability and Rehabilitation Research (NIDRR) for a Rehabilitation Engineering Research Center on Rehabilitation Robotics (RERC) at the Alfred I. duPont Hospital in Delaware from 1993–1997. With its ability to fund dozens of research projects in parallel, it also formed a partnership with a local company, Applied Resources, Corp. (ARC), which developed and marketed several rehabilitation technology products. One of the RERC researchers,

Rich Mahoney, moved to ARC and was instrumental in extending the company's repertoire to the RAPTOR wheelchair-mounted arm [53.18].

In Europe, the most significant mobile manipulator project was the MANUS project [53.19] mentioned earlier. With much of the work done under the direction of Hok Kwee at the Rehabilitation Research and Development Center (iRV) in the Netherlands, the project culminated in a robot specifically designed for wheelchair mounting, with control through a joystick and feedback by a small display on the arm itself. This project has led to numerous follow-on research projects, and, most significantly, to the commercialization of the system by Exact Dynamics BV, in the Netherlands. It is currently offered free on physician prescription by the Dutch government to qualified people with a disability such as cerebral palsy or tetraplegia from a spinal cord injury.

Autonomous navigation systems on electric wheelchairs also began in the 1980s, benefiting initially from the development by Polaroid Corporation of range finders for its cameras using ultrasonic sensors. They were inexpensive, and small enough, at 30 mm in diameter, that dozens of them could be placed around the periphery of a wheelchair to aid medium-range navigation (≈ 10 –500 cm). In the 1990s and early 2000s, with the advent of vision-based servoing and laser range scanners, algorithms for faster, smarter, less-error-prone navigation and obstacle avoidance dominated research advances in this sector. In Korea, for example, *Zenn Bien* at the Korea Advanced Institute for Science and Technology (KAIST) Human Welfare Robotics Center began developing the KAIST rehabilitation engineering system (KARES) line of wheelchair-based navigation systems in the late 1990s [53.20] and the NavChair project at the University of Michigan was the start of a development line that led to the commercialized Hephaestus system at the University of Pittsburgh [53.21, 22].

Therapy robots had a later start than assistive robots, with early exercise devices such as the BioDex [53.23] a first step in programmable, force-controlled, albeit single-axis devices, in the mid 1980s. The first multi-axis concept was published by *Khalili and Zomlefer* [53.24], and the first tested system by Robert Erlandson at Wayne State University emerged in the mid 1980s as well [53.25]. The RTX manipulator had a touch-sensitive pad as an end-effector, presenting targets at different locations for patients with upper-extremity weakness (e.g., following a stroke) to hit after the screen gave a visual signal. Software logged response times, thereby providing a score that was tallied and

compared to previous sessions. Later robots used advanced force-based control, which required significantly more computer power. The early 1990s saw the start of the MIT-MANUS Project with Neville Hogan and Igo Krebs, followed a few years later by the Palo Alto VA mirror image movement enabler (MIME) project and its derivative, Driver's simulation environment for arm therapy (SEAT), with Charles Bugar, Machiel Van der Loos, and Peter Lum, as well as the Rehabilitation Institute of Chicago ARM project with Zev Rymer and David Reinkensmeyer. Each had a different philosophy on upper-extremity stroke therapy and each was able to demonstrate clinical effectiveness in a different way. All three programs, now a decade later, have made significant technical advances and are still active.

Cognitive robotics had a start in the early 1980s to aid children with communication disorders and physical disorders to achieve a measure of control of their physical space. Using mostly educational manipulators,

several demonstration systems were developed. In the early 2000s, Corinna Latham of Anthrotronix, Inc. commercialized small robot systems to enable children with physical disabilities to play games with simple interfaces. Later, small mobile robots were used in clinics by *Kerstin Dautenhahn's* group [53.26] with children who have autism; since robots have such simple interfaces, communication with them does not appear not be as challenging as with other humans. The early 2000s also saw the advent of pet robots, such as the Paro seal robot developed by *Shibata et al.* [53.27], as companions for both children and the elderly who are confined to clinics and have limited real companionship.

The applications for robotics continue to increase in number as advances in materials, control software, higher robustness and the diminishing size of sensors and actuators allow designers to attempt new ways of using mechatronics technology to further the well-being of people with disabilities.

53.2 Physical Therapy and Training Robots

53.2.1 Grand Challenges and Roadblocks

The human neuromuscular system exhibits use-dependent plasticity, which is to say that use alters the properties of neurons and muscles, including the pattern of their connectivity, and thus their function [53.28–30]. The process of neuro-rehabilitation seeks to exploit this use-dependent plasticity in order to help people relearn how to move following neuromuscular injuries or diseases. Neuro-rehabilitation is typically provided by skilled therapists, including physical, occupational and speech therapists. This process is time-consuming, involving daily, intensive movement practice over many weeks. It is also labor-intensive, requiring hands-on assistance from therapists. For some tasks, such as teaching a person with poor balance and weak legs to walk, this hands-on assistance requires that the therapist have substantial strength and agility.

Because neuro-rehabilitation is time- and labor-intensive, in recent years health care payers have put limits on the amount of therapy that they will pay for, in an effort to contain spiraling health care costs. Ironically, at the same time, there has been increasing scientific evidence that more therapy can in some cases increase movement recovery via use-dependent plasticity. As robotics and rehabilitation researchers began to recog-

nize beginning in the late 1980s, neuro-rehabilitation is a logical target for automation because of its labor-intensive, mechanical nature, and because the amount of recovery is linked with the amount of repetition. Robots could deliver at least the repetitive parts of movement therapy at lower cost than human therapists, allowing patients to receive more therapy.

The grand challenge for automating movement therapy is determining how to optimize use-dependent plasticity. That is, researchers in this field must determine what the robot should do in cooperation with the patient's own movement attempts in order to maximally improve movement ability. Meeting this challenge involves solving two key problems: determining appropriate movement tasks (what movements should patients practise and what feedback should they receive about their performance), and determining an appropriate pattern of mechanical input to the patient during these movement tasks (what forces should the robot apply to the patient's limb to provoke plasticity). The prescription of movement tasks and mechanical input fundamentally constrains the mechanical and control design of the robotic therapy device.

There are two main roadblocks to achieving the grand challenge. The first is a scientific roadblock: neither the optimal movement tasks nor the optimal

mechanical inputs are known. The scientific basis for neuro-rehabilitation remains ill-defined, with competing schools of thought. The number of large, randomized, controlled trials that have rigorously compared different therapy techniques is still small, in part because these trials are expensive and difficult to control well. Therefore, the first problem that a robotics engineer will encounter when setting out to build a robotic therapy device is that there is still substantial uncertainty as to what exactly the device should do.

This uncertainty corresponds to an opportunity to use robotic therapy devices as scientific tools themselves. Robotic therapy devices have the potential to help identify what exactly provokes plasticity during movement rehabilitation, because they can provide well-controlled patterns of therapy. They can also simultaneously measure the results of that therapy. Better control over therapy delivery and improved quantitative assessment of patient improvement are two desirable features for clinical trials that have often been lacking in the past. Recent work with robotic movement training devices is leading, for example, to the characterization of computations that underlie motor adaptation, and then to strategies for enhancing adaptation based on optimization approaches [53.5, 31].

The second roadblock is a technological one: robotic therapy devices often have as their goal to assist in therapy of many body degrees of freedom (e.g., the arms and torso for reaching, or the pelvis and legs for walking). The devices also require a wide dynamic bandwidth such that they can, for example, impose a desired movement on a patient who is paralyzed, but also *fade-to-nothing* as the patient recovers. Furthermore, making the devices light enough to be wearable is desirable, so that the patient can participate in rehabilitation in a natural setting (for example, by walking over ground or working at a counter in a kitchen), or even throughout the course of normal activities of daily living. The development of high-degree-of-freedom, wearable, high-bandwidth robotic exoskeletons is an unsolved problem in robotics. No device at present comes close to matching the flexibility of a human therapist, in terms of assisting in moving different body degrees of freedom in a variety of settings (e.g., walking, reaching, grasping, neck movement), or the intelligence of a human therapist, in terms of providing different forms of mechanical input (e.g., stretching, assisting, resisting, perturbing) based on a real-time assessment of the patient's response. Meeting the grand challenge of robotic therapy therefore will require substantial, interrelated advances in both clinical neuroscience and robot engineering.

53.2.2 Movement Therapy after Neurologic Injury

At present, much of the activity in physical therapy and training robots has been focused on retraining movement ability for individuals who have had a stroke or spinal cord injury (SCI). The main reasons for this emphasis are that there are a relatively large number of patients with these conditions, the rehabilitation costs associated with them are high, and because these patients can sometimes experience large gains with intensive rehabilitation because of use-dependent plasticity. Some systems have also been targeted at assisting in cognitive rehabilitation of persons with neurologic injury, as reviewed below.

A stroke refers to an obstruction or breakage of a blood vessel supplying oxygen and nutrients to the brain. Approximately 800 000 people suffer a stroke each year in the USA alone, and about 80% of these people experience acute movement deficits [53.32]. There are over 3 000 000 survivors of stroke in the USA, with over half of these individuals experiencing persistent, disabling, movement impairments. The number of people who have experienced and survived a stroke is expected to increase substantially in the USA and other industrialized countries in the next two decades, because age is a risk factor for stroke and the mean age of people in industrialized countries is rapidly increasing due to the baby boom of the 1950s.

Common motor impairments that result from stroke are hemiparesis, which refers to weakness on one side of the body; abnormal tone, which refers to an increase in the felt resistance to passive movement a limb; impaired coordination, which can manifest itself as an apparent loss in control degrees of freedom and decreased smoothness of movement; and impaired somatosensation, which refers to a decreased ability to sense the movement of body parts. Secondary impairments include muscle atrophy and disuse-related shortening and stiffening of soft tissue, resulting in decreased passive range of motion of joints. Often the ability to open the hand, and to a slightly lesser extent close the hand, is dramatically decreased.

The number of people who experience a SCI in the USA each year is relatively smaller – about 15 000, with about 200 000 people alive who have survived a SCI – but the consequences can be even more costly than stroke [53.32]. The most common causes of SCI are automobile accidents and falls. These accidents crush the spinal column and contuse the spinal cord, damaging or destroying neurons within the spinal cord. The resulting pattern of movement impairment depends strongly on

the vertebrae at which the spinal cord is injured, since nerves branch out to the head, arms, legs, and bladder and bowel at different vertebrae. About 50% of spinal cord injuries are incomplete, meaning that some sensation or motor function is preserved below the level of the injury. Spinal cord injuries are commonly bilateral and thus are often more functionally devastating in comparison to strokes, which at least leave a person with one side of their body that is relatively normal (which we will refer to as the *less impaired* side). Individuals experience especially severe disability if the lesion is high enough to involve the arms as well as the legs.

53.2.3 Robotic Therapy for the Upper Extremity

The following sections describe the best-known clinically tested upper-limb therapy robot systems that have been developed since the 1980s (Fig. 53.1).

MIT-MANUS

The first robotic therapy device to undergo extensive clinical testing, and now to achieve some commercial success, is the MIT-MANUS, sold as the InMotion2 by Interactive Motion, Inc. [53.33]. MIT-MANUS is a planar two-joint arm that makes use of the selective compliant assembly robot arm (SCARA) configuration, allowing two large, mechanically grounded motors to drive a lightweight linkage. The patient sits across from the device, with the weaker hand attached to the end-effector, and the arm supported on a table with a low-friction support. By virtue of the use of the SCARA configuration, the MIT-MANUS is perhaps the simplest possible mechanical design that allows planar movements while also allowing a large range of forces to be applied to the arm without requiring force feedback control.

MIT-MANUS assists the patient in moving the arm across the tabletop as the patient plays simple video

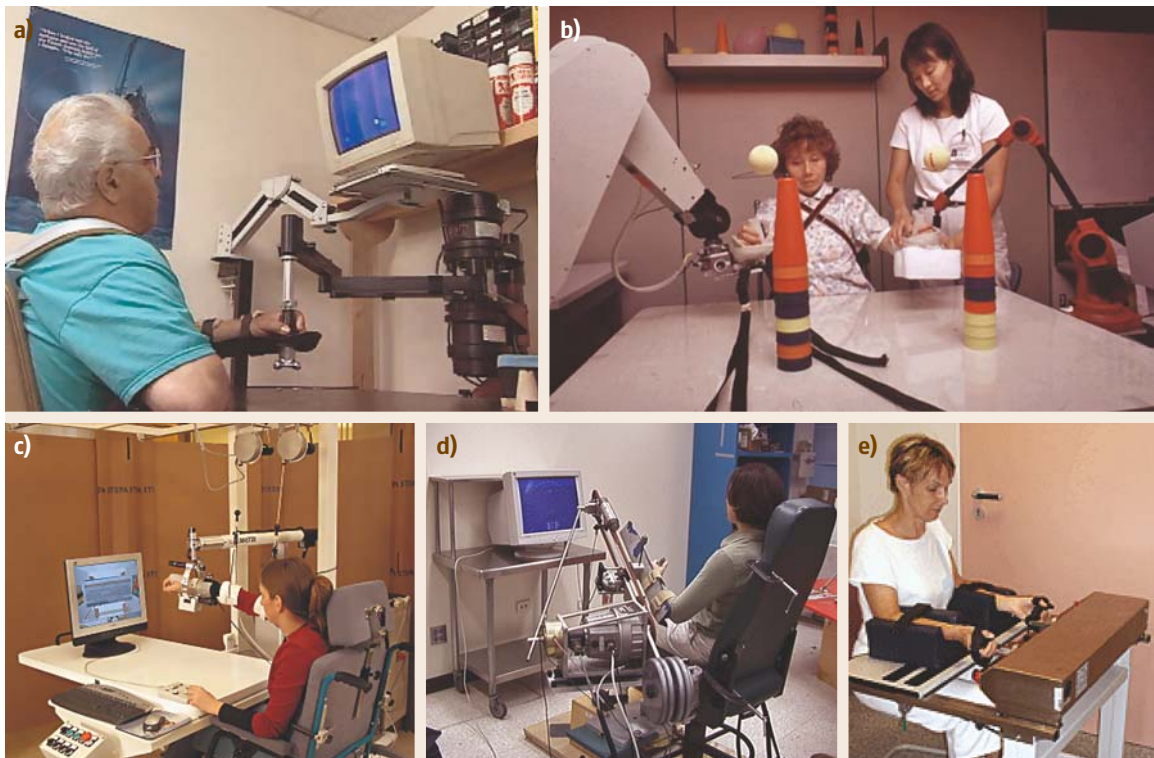


Fig. 53.1a–e Arm-therapy robotic systems that have undergone extensive clinical testing; (a) MIT-MANUS, developed by Hogan, Krebs, and colleagues at the Massachusetts Institute of Technology (USA); (b) MIME, developed at the Department of Veterans Affairs in Palo Alto in collaboration with Stanford University (USA); (c) GENTLE/s developed in the EU, (d) ARM-Guide, developed at the Rehabilitation Institute of Chicago and the University of California, Irvine (USA), and (e) Bi-Manu-Track, developed by Reha-Stim (Germany)

games, such as moving a cursor into a target that changes locations on a computer screen. Assistance is achieved using a position controller with an adjustable impedance. Additional modules have been developed for the device for allowing vertical motion [53.34], wrist motion [53.35], and hand grasp [53.36]. Software has been developed for providing graded resistance as well as assistance to movement [53.37], and for varying the firmness and timing of assistance based on real-time measurements of the patient's performance on the video games [53.38].

MIT-MANUS has undergone extensive clinical testing in several studies, summarized as follows. The first clinical test of the device compared the motor recovery of acute stroke patients who received an additional dose of robot therapy on top of their conventional therapy, to that of a control group, who received conventional therapy and a brief, sham exposure to the robot [53.39]. The robot group patients received the additional robotic therapy for an hour each day, five days per week, for several weeks. The robot group recovered more arm movement ability than the control group according to clinical scales, without any increase in adverse effects such as shoulder pain. The improvements might subjectively be characterized as *small but somewhat meaningful to the patient*. The improvements were sustained at three-year follow-up.

This first study with MIT-MANUS demonstrated that acute stroke patients who received more therapy recover better, and that this extra therapy can be delivered by a robotic device. It did not answer the question as to whether the robotic features of the robotic device were necessary. In other words, it may have been that patients would have also improved their movement ability if they had practised additional movements with MIT-MANUS with the motors off (thus making it equivalent to a computer mouse), simply by virtue of the increased dose of movement practice stimulating use-dependent plasticity. Thus, while this study indicated the promise of robots for rehabilitation therapy, it did not close the gap of knowledge as to how external mechanical forces provoke use-dependent plasticity.

Subsequent studies with MIT-MANUS confirmed that robotic therapy can also benefit chronic stroke patients [53.40]. The device has been used to compare two different types of therapy – assisting movement versus resisting movement – in chronic stroke subjects, but with inconclusive results: both types of therapy produced benefits [53.37]. The device has also been used to compare assistive robot therapy with another technological approach to rehabilitation – electrical stimulation of

finger and wrist muscles [53.41]. Again, significant benefits were found for both therapies, and those benefits were specific to the movements practised, but the benefits were not significantly different between therapies. We note that the lack of a significant difference in these studies may simply be due to the limited number of patients who participated in these studies (i.e., inadequate study power), rather than a close similarity of the effectiveness of the therapies.

MIME

Several other systems have undergone clinical testing. The mirror image movement enhancer (MIME) system uses a Puma-560 robot arm to assist in movement of the patient's arm [53.42]. The device is attached to the hand through a customized splint and a connector that is designed to break away if interaction forces become too large. Compared to MIT-MANUS, the device allows more naturalistic motion of the arm because of its six degrees of freedom (DOFs), but must rely on force feedback so that the patient can drive the robot arm. Four control modes were developed for MIME. In the passive mode, the patient relaxes and the robot moves the arm through a desired pattern. In the active assist mode, the patient initiates a reach toward a target, indicated by physical cones on a table top, which then triggers a smooth movement of the robot toward the target. In the active constrained mode, the device acts as a sort of virtual ratchet, allowing movement toward the target, but preventing the patient from moving away from the target. Finally, in mirror-image mode, the motion of the patient's less impaired arm is measured with a digitizing linkage, and the impaired arm is controlled to follow along in a mirror-symmetric path. The initial clinical test of MIME found that chronic stroke patients who received therapy with the device improved their movement ability about as much as patients who received conventional tabletop exercises with an occupational therapist [53.42]. The robot group even surpassed the gains from human-delivered therapy for the outcome measures of reaching range of motion and strength at key joints of the arm. A follow-on study is being undertaken to elucidate which of the control modes or what combination of MIME exercises caused the gains [53.43].

ARM Guide

The question of the effect of robot forces on movement recovery was also left unresolved by a study with another device, the ARM guide. The ARM guide is a trombone-like device that can be oriented then locked

in different directions and assist in reaching in a straight line. Chronic stroke patients who received assistance during reaching with the robot improved their movement ability [53.44]. However, they improved about as much as a control group that simply practised a matched number of reaches without assistance from the robot. This suggests that movement effort by the patient is a key factor for recovery, although the small sample size of this study limited its ability to resolve the size of the difference between guided and unguided therapy.

Bi-Manu-Track

Perhaps the most striking clinical results generated so far have come from one of the simplest devices built. Similar to a design proposed previously [53.45], the Bi-Manu-Track uses two motors, one for each hand, to allow bimanual wrist-flexion/extension [53.46]. The device can also assist in forearm pronation/supination if it is tilted downward and the handles are changed. In an extensive clinical test of the device, 22 subacute patients (i. e., 4–6 weeks after stroke) practised 800 movements with the device for 20 min per day, five days per week for six weeks [53.46]. For half of the movements, the device drove both arms, and for the other half, the patient's stronger arm drove the motion of the more-impaired arm. A control group received a matched duration of electrical stimulation (ES) of their wrist extensor muscles, with the stimulation triggered by voluntary activation of their muscles when possible, as measured by electromyography (EMG). The number of movements performed with EMG-triggered ES was 60–80 per session. The robot-trained group improved by 15 points more on the Fugl-Meyer scale, a standard clinical scale of movement ability with a range from 0 to 66 points in upper extremity function. It assigns a score of 0 (cannot complete), 1 (completes partially), or 2 (completes normally) for 33 test movements, such as lifting the arm without flexing the elbow. For comparison, reported gains in Fugl-Meyer score after therapy with the MIT-MANUS and MIME devices ranged from 0–5 points [53.47].

Other Devices to Undergo Clinical Testing

Other devices to undergo clinical testing are as follows. The GENTLE/s system uses a commercial robot, the HapticMaster, to assist in patient movement as the patient plays video games. The HapticMaster allows four degrees of freedom of movement and achieves a high bandwidth of force control using force feedback. Chronic stroke patients who exercised with GENTLE/s improved their movement ability [53.48, 49]. The Rutgers hand robotic device uses low-friction pneumatic

cylinders to help extend or flex the fingers, and has been shown to improve hand movement ability of chronic stroke subjects [53.50]. Simple force-feedback controlled devices, including a one-DOF wrist manipulator and a two-DOF elbow–shoulder manipulator, were also recently shown to improve movement ability of chronic stroke subjects who exercised with the devices [53.51]. A passive exoskeleton, the T-WREX arm orthosis, provides support to the arm against gravity using elastic bands, while still allowing a large range of motion of the arm [53.52]. By incorporating a simple hand-grasp sensor, this device allows substantially weakened patients to practise simple virtual reality exercises that simulate functional tasks such as shopping and cooking. Chronic stroke patients who practised exercising with this non-robotic device recovered significant amounts of movement ability, comparable with the Fugl-Meyer gains seen with MIT-MANUS and MIME. NeReBot is a three-DOF wire-based robot that can slowly move a stroke patient's arm in spatial paths. Acute stroke patients who received additional movement therapy beyond their conventional rehabilitation therapy with NeReBot recovered significantly more movement ability than patients who received just conventional rehabilitation therapy [53.53]. RehaRob uses an industrial robot arm to mobilize patients' arms along arbitrary trajectories following stroke [53.54].

Other Systems Currently under Development

Several other robotic therapy devices are currently under development. For example, at the high end of cost and complexity are the ARM-In [53.55] and Pneu-WREX systems [53.56], which are exoskeletons that accommodate nearly naturalistic movement of the arm while still achieving a wide range of force control. A system that couples a immersive virtual-reality display with a haptic robot arm is described in [53.57]. A wearable exoskeleton driven by pneumatic muscles is described in [53.58]. At the lower end of cost/complexity are devices that use force feedback joysticks and steering wheels with a view toward implementation in the home [53.59–62]. Examples of recent, novel robotic devices for the hand are given in [53.63–65]: these devices typically follow an active assist therapy paradigm in that they are designed to help open and close the hand. One robotic therapy system for the hand incorporates the idea of using visual feedback distortion to enhance motivation of patients during movement therapy [53.66]. Using robotic force fields to amplify the kinematic errors of stroke patients during reaching may provoke novel forms of adaptation of those patterns [53.4, 67].

53.2.4 Robotic Therapy for Walking

Background

Scientific evidence that gait training improves recovery of mobility after neurologic injury started to accumulate in the 1980s through studies with cats. Cats with SCI can be trained to step with their hind limbs on a treadmill with partial support of the body weight and assistance of leg movements [53.68, 69]. Following the animal studies, various laboratories developed a rehabilitation approach in which the patient steps on a treadmill with the body weight partially supported by an overhead harness and assistance from up to three therapists [53.70–73]. Depending on the patient's impairment level, from one to three therapists are needed for body-weight supported treadmill training (BWSTT), with one therapist assisting in stabilizing and moving the pelvis, while two additional therapists sit next to the treadmill and assist the patient's legs in swing and stance. This type of training is based on the principle of generating normative, locomotor-like sensory input that promotes functional reorganization and recovery of the injured neural circuitry [53.74]. In the 1990s, several independent studies indicated that BWSTT improves stepping in people with SCI or hemiplegia after stroke [53.70–72].

Gait training is particularly labor-intensive and strenuous for therapists, so it is an important target for

automation. The efforts of roboticists have been especially focused on BWSTT rather than overground gait training because BWSTT is done on a stationary setup in a well-defined manner and thus can be more easily automated than overground gait training. Randomized, controlled clinical trials have shown that BWSTT is comparable in effectiveness to conventional physical therapy for various gait-impairing diseases [53.75–80]. These trials support the efforts towards automation of BWSTT, as the working conditions of physical therapists will improve if the robots do much of the physical work, which in the case of BWSTT actually leads to occasional back injuries to therapists. Usually, only one therapist is needed in robot-assisted training, for the tasks of helping the patient into and out of the robot and monitoring the therapy. In the case of SCI patients, a small randomized, controlled trial [53.76] reported that robotic-assisted BWSTT with a first-generation robot required significantly less labor than both conventional overground training and therapist-assisted BWSTT, with no significant difference found in effectiveness.

Gait-Training Robots in Current Clinical Use

Three gait-training robot systems are already in use for therapy in several clinics worldwide: the gait trainer GT-I [53.81], the Lokomat[®] [53.82], and the AutoAmbulator[™] [53.83] (Fig. 53.2).



Fig. 53.2a–c Gait-training robotic systems currently in use in clinics; **(a)** the gait trainer GT-I, developed by Hesse's group and commercialized by Reha-Stim (Germany); **(b)** the Lokomat[®], developed by Colombo and colleagues and commercialized by Hocoma AG (Switzerland), and **(c)** AutoAmbulator[™], developed by the HealthSouth Corporation (USA)

Of these three robot systems, the GT-I is the one that departs most from therapist-assisted BWSTT, since it interacts with the patient's lower limbs through two footplates rather than acting on the shank as human therapists do. It also appears to depart more from natural walking because the footplate principle substantially alters the sensory cues of the foot impact with the ground or treadmill band. The GT-I footplates are driven by a singly actuated mechanism that moves the foot along a fixed gait-like trajectory with a doubled crank and rocker system [53.81]. The stride length can be adjusted between sessions by changing gears. The body weight is unloaded as needed by an overhead harness. The torso is moved sagittally in a phase-dependent manner by ropes attached to the harness and connected by another crank to the foot crank. The GT-I is currently installed in dozens of clinics, mainly in Europe. One randomized, controlled study has been reported that tested the GT-I with 30 subacute stroke patients [53.84]. The robot group improved their overground walking ability more than the control group, although differences were not significant at six-month follow-up. A total of 80% of the patients said they preferred training with the robot rather than the therapists because training with the robot was less demanding and more comfortable. The other 20% of patients stated that swinging of the paretic limb seemed less natural and thus less effective when training with the robot. Robot-assisted training required an average of one therapist per patient, while therapist-assisted training required two therapists per patient on average. A follow-up, randomized controlled study comparing conventional training plus robotic training with the GT-I with a time-matched amount of conventional training alone with subacute stroke patients found that the group that received some robotic training recovered walking ability to a great extent [53.85].

The Lokomat is a robotic exoskeleton worn by the patients during treadmill walking [53.82]. Four motorized joints (two per leg) move the hip and knee. The actuators consist of ball screws connected to direct-current (DC) motors. The legs are driven in a gait-like pattern along a fixed position-controlled trajectory. The device attaches to the thighs and shanks through padded straps. A passive parallelogram mechanism allows vertical translation of the patient's torso, restricting lateral translation. The patient's body weight is unloaded as needed through an overhead harness. The Lokomat is currently being used in dozens of research labs and clinics worldwide. In 2005, Wirz and coworkers [53.86] reported preliminary results of robot-assisted BWSTT

with the Lokomat in 20 chronic incomplete SCI patients. The improvements in overground walking speed and endurance were statistically significant: approximately 50% gain on average in the 16 patients who were ambulatory before training. There were no significant changes in the requirement of walking aids, orthoses or external physical assistance. The improvements appear to be comparable to those achieved by similar SCI patients who received therapist-assisted BWSTT. Hornby and colleagues [53.76] studied the effects of robot-assisted BWSTT with the Lokomat on individuals with subacute SCI. Thirty patients were randomly assigned to one of three training groups: robot-assisted BWSTT, therapist-assisted BWSTT, and therapist-assisted overground ambulation. Improvements in motor and functional abilities were similar in the three training groups, with the robot-assisted BWSTT requiring significantly less labor than the other two therapy methods.

The AutoAmbulator (<http://www.autoambulator.com>) consists of two robotic arms that assist patients to step on a treadmill with their body weight supported as needed. The interface to the patient's legs is through straps at the thigh and ankle. The AutoAmbulator is currently being used in 57 HealthSouth rehabilitation centers, all of them in the United States. A single-blind, randomized clinical trial to assess its effectiveness in stroke patients is currently underway.

Further Research and Development on Robotic Therapy for Walking

Several groups worldwide are working toward improving gait-training robotic technologies. A great deal of effort has been going into incorporating and investigating the ability to assist as needed [53.31, 87–91], that is, the ability of the robot to let the patients contribute to the locomotor efforts as much as they are able. This is likely essential for maximizing locomotor plasticity [53.92]. Some effort has also been directed towards adding more active DOF, particularly for torso manipulation [53.90, 93]. These robotic tools are needed not only for their potential clinical use in therapy, but for studying what aspects of the assistance are important for effective gait training and how best to control and implement them with robotic devices.

The team responsible for the GT-1 has developed the HapticWalker [53.87], which maintains the permanent foot/machine contact but allows the footplates to move along three-DOF trajectories. In addition it incorporates force feedback and compliance control, as well

as haptic simulation of ground conditions (e.g., stair climbing).

An advanced version of the Lokomat integrates force sensors and automatic adaptation of gait patterns to allow for a reduction of the interaction effort between patient and machine [53.88]. It has been tested on unimpaired and SCI patients, who were able to influence the gait trajectories towards a more desired motion by means of their own motor activity [53.88,94].

The pelvic assist manipulator (PAM) is a five-DOF robot for torso manipulation, and the pneumatically operated gait orthosis (POGO) is a leg robot with two DOFs per leg. PAM's and POGO's actuators are pneumatic, which cost less than electric motors and have higher power-to-weight ratios [53.93]. The robots' ability to control forces and yield to patients and/or therapists has been tested with unimpaired and SCI patients [53.95]. Of particular note here is the development of an adaptive synchronization algorithm that allows these compliant robots to provide assistance at the right time as the patient varies the timing or size of steps.

Based on the string-puppet principle, the String-Man achieves weight bearing and compliant six-DOF torso manipulation by means of seven wires and a force sensor on each wire [53.90]. In addition, a control scheme has been designed for the String-Man to control both the zero-moment-point location and the ground reaction force with the help of foot force sensors.

Veneman and colleagues [53.91] are developing actuation systems for robotic exoskeletons that combine Bowden cables with series elastic actuation. The Bowden cables allow the motors to be mounted remotely in a fixed position, thus reducing the mass to be moved on the exoskeleton links. The spring element connecting the Bowden cables with the joint allows the closing of a force feedback control loop with a position sensor that measures the spring elongation, a concept inspired from the series-elastic actuation concept described by Pratt and coworkers [53.96]. In addition, series elasticity is useful to reduce the negative effects that static friction and unmodeled dynamics have on the stability of force control, thus improving the achievable force control bandwidth [53.97,98], which is especially important for a Bowden-cable-based system. Series elasticity worsens positional accuracy, but this is not a critical issue for gait-training robots. Veneman's first experimental results show that adequate force control bandwidth was achieved by a prototype of their Bowden-cable-based actuation design [53.91], so that a robot incorporat-

ing this actuation concept can plausibly execute both a stiff, position-dominated robot-in-charge mode as well as a compliant, low-impedance patient-in-charge mode.

A different approach to gait training was taken with the KineAssist device [53.99]. KineAssist is a motorized mobile platform that follows patient and therapist as they move over ground and incorporates a smart brace that compliantly supports the patient's trunk and pelvis. This smart support is designed to allow the therapist to adjust its stiffness from fully rigid down to fully compliant. Within a safety zone, the fully compliant mode allows patients to challenge the limits of their stability. A compliant virtual wall catches the patients when they lose balance. The location of this virtual wall is also adjustable. The body weight can be unloaded as needed.

Other efforts include *Ferris* and coworkers [53.100], who are developing foot, ankle, knee, and hip orthoses actuated by artificial pneumatic muscles that may possibly be used to assist in gait training. The Rutgers Ankle is a six-DOF pneumatic system based on a Stewart platform that allows exercise of the ankle [53.101]. Also in the USA, *Agrawal's* group proposes the use of gravity-balancing leg orthoses for people with gait impairments to practise walking [53.102]. Their designs allow the orthoses to passively support the gravity torque required at the patient's joints. This approach would have the advantage of being safer than powerful robots for clinical use. They have also extended their design to include actuators with reduced torque requirements [53.103]. A robot has been used to provide graded body weight support as a patient who cannot bear his full weight because of a medical problem walks in a circle [53.104].

Other Robotic Movement Therapy Approaches

As reviewed above, most of the work to date in robotic therapy devices has focused on robots that attach to patients to assist them in practising reaching or walking exercises. Other early proposals for using robots for movement therapy included using two planar robot arms to carefully control continuous passive motion of the knee following joint surgery [53.24], and using a multi-axis robot arm to place targets for patients doing reaching exercises [53.105]. An emerging approach toward robotic movement therapy is to provide the therapy at a distance, in a form of tele-rehabilitation, in order to improve accessibility to the therapy [53.59,106,107].

53.3 Aids for People with Disabilities

53.3.1 Grand Challenges and Enabling Technologies

Enabling technologies assist people with disabilities to achieve a quality of life on a par with able-bodied individuals through increased functional independence. The main issue with most such technologies is that disability has a highly individualized impact: a solution for one person will not work for someone else, even if their disabilities appear clinically similar. The more a disability impacts function, the more costly the technical intervention tends to be, since the consumer market cannot benefit from economies of scale if each solution must be individualized. As an extreme example, an electric wheelchair with individualized padding, motorized recliner, and customized joystick control costs as much as a mass-produced mid-sized automobile, but has a fraction of the electronics, robustness, and functions. A grand challenge for assistive, enabling technologies is to find a means to make mass-personalization possible, as it has been in the automotive industry, for example. One component is designer focus. If we can re-badge *assistive technology* as *design for well-being products*, the change in focus from *fixing* people to improving their quality of life will have the effect of mainstreaming disability itself so that manufacturers of consumer equipment tend to develop products that can explicitly accommodate a much wider range of functional abilities and therefore provide benefit to a larger, overall less-able, consumer base. As the average age of the baby boomers climbs into retirement years with significant disposable income, this segment will compel the market into providing better solutions to their well-being needs.

Another grand challenge is robotic autonomy. Especially for persons with reduced communication, physical, and/or cognitive abilities, a rehabilitation robot will need to have sensory (e.g., vision, auditory) and motor capabilities, combined with its own software processing capabilities (also termed artificial intelligence), that make it a sufficiently safe and capable system to coexist with and benefit humans. This challenge will to some extent be dependent on continuing increases in computer processing power, but also specifically dependent on the algorithmic developments that issue from the community of robotics researchers.

Research on robotic aids has so far primarily targeted persons with mobility and manipulation limitations, rather than children and adults with cognitive impairments. However, increases in the prevalence of cognitive

impairments related to aging will make the latter focus increasingly important. Research has been limited to the mobility focus due to the difficulty of designing and developing intrinsically safe robots that can coexist with people and exhibit a certain amount of autonomy while performing useful work. Robots today therefore rely on user vigilance and explicit control to be safe. If the user does not have the cognitive capacity to evaluate a robot's safety situation or the ability to communicate efficiently, then the positive value of a function-enhancing robot is nullified by the harm that it could inflict on the user or bystanders. Coupled with the fact that the design of interfaces to personal robots is still in its infancy, a challenge for robotic aid developers is a significant improvement in intrinsic safety without a decrease in function (strength, speed, etc.) from what is typical today in industrial robotics.

Disabilities and Functional Limitations Served by Robotic Aids

Assistive robots have been designed for people who have become severely disabled as a result of, for example, muscular dystrophy or a high-level SCI, for children who have cerebral palsy (CP), and more generally for anyone who lacks the ability to manipulate household objects. A market research study conducted ten years ago specifically for rehabilitation robotics clients conservatively projected a US market of 100 000 people [53.108]. With the incidence of disability increasing exponentially, and the niches that robots can fill in rehabilitation applications multiplying with advances in robotics and rehabilitation science, it is clear that the market for rehabilitation robotics can only continue to increase.

Human–Robot Interface Design for Assistive Robots

A fundamental difference between using industrial and assistive robots is the interface required to command, control, and ultimately benefit from them. An industrial robot commonly has a combination of a manual controller and a programming language interface to allow an operator to teach a robot where to go and to enter the specific motion, grasping, tool changing, and error-recovery steps it must follow repeatedly in its factory automation scenario. A rehabilitation robot, on the other hand, typically has three main differences and challenges: (1) the operator is not by definition a roboticist or engineer, so the interface must make accessible all the functions of the robot to allow its user to complete the required tasks;

(2) the user of a rehabilitation robot is, by definition, a person with a disability, which means that physical, sensory, communication, and/or cognitive limitations in accessing the commands and controls of a robot need to be handled on a systems level by the designers of robots and their interfaces, with critical attention to universal design principles; and (3) all rehabilitation robots require individualization of the interface to each user by the engineering and therapy professionals in charge of prescription and fitting, since disabilities vary considerably in how they restrict adaptability to standard configurations [53.109].

Interfaces of assistive robots tap into the residual communication capabilities of each user. For example, many people with tetraplegia retain the ability to move a hand, arm, foot or the head in a repeatable even if range-limited way, and possibly even in two axes, such as forward–backwards and left–right. With proper placement of push buttons, a joystick, or non-contact position measurement device, a rehabilitation engineer and therapist can develop a custom solution for each of their clients with disabilities to control a wheelchair computer and robot. In addition, adaptive hardware and software for control of a computer, such as head-position cursor control, eye-trackers, speech recognition systems, track balls, and special keyboards, can be used to provide access to computer-based robot functionality.

Even more so than for able-bodied computer and robot users, redundancy in input modality is important for persons with disabilities to prevent a system from becoming inoperable due to a simple interface malfunction or calibration problem. Providing two means of creating a mouse click action (for example, a separate button placed next to a cursor-control track ball, as well as dwell time on a software *button* on-screen), even if one is inherently slower than the other, allows continued and uninterrupted use of the computer without outside assistance even if one of the two fails.

For therapy robots, physical interfaces resemble those for physical and occupational therapy equipment in general and have a commonality with sports equipment interfaces as well, with adjustable hook-and-loop-type straps, heat-formable plastic cuffs, soft rubber, foam-based materials, and durable coverings for abrasion resistance and long wear. After a session or two for fitting and adaptation, a person using a therapy robot can often use the same interface for a long period of time.

In summary, the keys to interface design are customizability, individualization, functional redundancy, adaptability, and patience in getting the interface to

a comfort and functional level appropriate for effective use of the robot.

53.3.2 Types and Examples of Assistive Rehabilitation Robots

As mentioned in the Introduction, assistive robots can be divided into three main categories: manipulation aids, mobility aids, and cognitive aids. Each can be subdivided as follows. Manipulation aids are commonly divided into fixed, portable, and mobile subtypes. Mobility aids are divided into electric wheelchairs with autonomous navigation features and smart walkers. Cognitive aids are divided into communication aids such as pet robots and autonomous caretaker robots. These categories are introduced below, and representative systems that have undergone scientific user studies or are commercial products are presented (Fig. 53.3). Other examples are mentioned in the history in Sect. 53.1.3.

Manipulation Aids: Fixed Base

Common robots of this type are **ADL** and vocational manipulation aids and kitchen robots. In the **US**, the professional vocational assistive robot (ProVAR) is a research prototype based initially on a PUMA-260 robot arm mounted on a 1 m transverse overhead track that allows the robot to manipulate objects and operate devices on side shelves and the tabletop, bringing objects (like a drink of water or throat lozenge) to the robot's operator. The interface is via a Java or virtual-reality modeling language (**VRML**) plug-in to a common Internet browser, delivering high-level control to disabled office workers in a conventional pull-down menu and control screen interface [53.14, 110]. This system and its predecessor **DeVAR** have been field tested by over 50 subjects at five rehabilitation clinics to assess feasibility and acceptability [53.111, 112]. At a cost of over **US\$** 100 000 currently, it is poised to be re-engineered with a simpler, cheaper arm for eventual product introduction.

In the European Union (EU), following a development path parallel to ProVAR's, is the **AfMASTER/RAID** workstation, whose concept, instead of being built into a workstation, includes a 2 m × 3 m robot work area in the user's office to store objects and place appliances, next to the user's own office space. The system has been developed over a 20 year span and is in limited production [53.16].

The kitchen robot **Giving-A-Hand**, developed at the Scuola Superiore Sant'Anna in Pisa, Italy, is a low-degree-of-freedom device for mounting on the front rail of a kitchen counter and able to move food contain-

ers to and from appliances such as refrigerators and ovens [53.113]. With an integrated control system, it can also make use of the internal controls of the devices to, for example, set cooking times and open doors.

The UK-developed Handy-1 is a domestic robot with three degrees of freedom designed for one-switch operation by persons with cerebral palsy [53.17]. Originally designed to allow a person to eat a meal one bite at a time, its application areas have been extended to face hygiene and cosmetics. A commercial product selling for about US\$ 6000, it has been a commercial success due to its simplicity and application focus. An even simpler feeding robot, the UK's electric Neater Eater [53.8] is on sale worldwide at about US\$ 5000, and is designed for eating only.

While a robot conventionally connotes a stand-alone system with some automation features, a smart bed and a smart home can legitimately be termed robots since they sense and act with motors under the shared control of its human users and its real-time software programming. Smart beds such as SleepSmart measures body position and temperature, as well as trends and anomalies over the course of a night. Restlessness can be measured, and bed geometry (tilt of bed segments) and ambient conditions (light, temperature, sounds) can be adjusted according to presets and preferences [53.114].

Smart homes, such as the domotic environment at Georgia Tech, NL-iRV, and the University of Tokyo [53.115], provide integrated climate, security, lighting, entertainment, and transport assistance, which is enabling especially to persons with severe functional disabilities. Coupled with health-care-related functionality (see the next section), these robotic homes can allow a person with a cognitive or physical disability to control many ADL functions and live safely through monitoring.

Manipulation Aids:

Wheelchair Manipulator Arm Systems

A need for electric wheelchair users is the manipulation of objects while navigating a home or a public place such as a restaurant or grocery store. The assistive robot service manipulator (ARM, Exact Dynamics, Netherlands) – previously known as MANUS – is a commercial robot arm that can be attached to an existing wheelchair to the side of the lap tray and controlled by the wheelchair's own joystick or a number pad [53.19, 116] (Fig. 53.4). The robot has undergone numerous user studies with persons who have muscular dystrophy, a high-level SCI, or cerebral palsy. Worldwide, this is currently the only commercial rehabilitation robot arm that can be prescribed



Fig. 53.3a–c Workstation-type robots: (a) AfMaster, developed by the French Muscular Dystrophy Association, (b) ProVAR, developed at the VA Palo Alto Rehabilitation R&D Center, and (c) Handy-1, developed by RehabRobotics, Ltd. (UK)

by a physician and that is reimbursed by a government health care system.

Manipulation Aids:

Mobile Autonomous Systems

The most commonly conceived form of a robot is that of an autonomous, mobile system with arms, having sensorimotor functionality similar to that of a human being, while serving people in performing menial physical tasks. Chapter 56 of this Handbook explores the domain as well. Since locomotion is a key requirement for humanoid robotics, other robots with wheeled bases have been developed before the first walking robots were invented to explore more applied domains with more short-term usefulness. In film, robots such as Star Wars' R2D2 have made this form factor commonly known



Fig. 53.4 Wheelchair manipulator robot MANUS developed at the Rehabilitation R&D Center, Hoensbroek, and marketed by Exact Dynamics (The Netherlands)

robot that can also be used as a physical support to people requiring mobility and stability assistance. It has also doubled as a mobile kiosk, moving around a trade show floor and delivering information to attendees.

Mobility Aids: Wheelchair Navigation Systems

A critical function for people who use electric wheelchairs for their mobility impairment and who in addition have communication or cognitive disability is semi-autonomous navigation assistance (Fig. 53.5). Add-ons to commercial wheelchairs have been developed by numerous research groups for this service. The NavChair [53.120] was one of the first to demonstrate robust wall-following, door passage even with narrow doorways, and speed adaptation to people walking in front of the wheelchair, all using only short-range ultrasonic and other sensors, but not vision. The Hephaestus [53.22] is a next-generation system made specifically as a commercial accessory for a variety of wheelchair brands, tapping into the joystick controller and power system. The Wheelseley [53.121] and KARES [53.20] robots have explored similar functionality using a vision system for scene analysis and way-finding.

Mobility Aids: Walking Assistance Systems

A third type of mobile robot for stability assistance has the particularity that it is underactuated and has similarity with a *co-bot* in that the wheels are not driven, but are actively steered and braked (Fig. 53.6). The Pam-Aid [53.122] looks like a closed-front walker on wheels and has bicycle-type handlebars. The person walking behind the device turns the handlebars, causing the wheels to turn in the correct direction. If the ultrasonic sensors detect an obstacle in front of it, the brakes prevent the user and device from colliding with it. The Care-O-bot (see before), designed originally as a mobile autonomous robot of approximately human size, has a similar set of handlebars to Pam-Aid so it can be used as a smart walker. The larger mass of the Care-O-bot, however, requires it to be motorized.

Cognitive Aids

There has recently been increased interest in using robots as motivational and educational agents during rehabilitation therapy. This approach typically involves small, pet-like, toy-like, approachable devices that do not physically interact with the patient, but exist primarily to engage the patient in an affective way that promotes personal health, growth and interaction. For example, the



Fig. 53.5a,b Wheelchair navigation aids: (a) Wheelseley and (b) Hephaestus

around the world. More recently, real robots such as the HelpMate [53.117] have been employed in US hospitals as fetch-and-carry robot orderlies, using floor maps and short-range ultrasonic sensors for navigation and obstacle avoidance. The Italian MovAid research robot platform [53.118] adds manipulation and vision to these capabilities to navigate in home-like environments to provide object manipulation and device operation to individual users. The German Care-O-bot [53.119] has explored advanced navigation and sensing in a wheeled



Fig. 53.6a–c Human assistance robots: (a) Care-O-bot, developed by the Fraunhofer Research Institute (Germany). (b) Helpmate by Transitions Research Inc., USA. (c) Pam-Aid (aka Guido), developed in the UK

PARO robot looks like a baby harp seal, and can respond to sound, contact, light, and tilt by moving its head and eyelids [53.123]. The device has been used to facilitate interaction with children with autism and elderly adults. The CosmoBot, a small humanoid robot with multiple degrees of freedom and a control panel, has been used to motivate speech and motor behavior by children with cerebral palsy, Down's syndrome, and autism [53.124]. The Japanese Wakamaru robot [53.125] was announced by Mitsubishi Heavy Industries in 2003 as a home health monitoring robot to assist persons living alone but at risk of falls and in need of information services and reminders related to their own health care. An interesting feature of this robot is that its arms and programmable facial features are used for gesturing only, not manipulation and sensing, to assist in effectively communicating to its users. A small, talking mobile robot was recently developed to motivate and measure rehabilitation exercises by stroke patients [53.126]. There is also extensive related work in the development of virtual-reality interfaces for motivating rehabilitation therapy [53.127].

As an example using robots to motivate therapy, Dautenhahn and colleagues have studied the use of robotic toys as tools to stimulate communicative and social skills and facilitate human contact in children with autism. Children and adults with autism often avoid social interactions and have difficulty interpreting facial expressions and other social cues in interactions with people. As reviewed by Dautenhahn [53.26], research

suggests that children with autism generally feel more comfortable in predictable environments. Human social behavior is subtle, elaborate, and unpredictable. Many children with autism are however interested to play with mechanical toys or computers. In addition, Dautenhahn's studies show that children with autism have greater interest in interactions with autonomous robotic toys than with inanimate toys. In a study spanning several months [53.128], four children with autism were comfortable interacting with a humanoid robotic doll called Robota, participating in imitation, turn-taking, and role-switching. The children even started, by their own initiative, to include the investigator in their shared experience with the robot. Robins et al. suggest that robots could be used to encourage social interactions by children with autism, to allow care-givers to observe and monitor the children's play behavior and progress and to address specific therapeutic goals by targeted programming.

Other cognitive aids are focused on caretaker tasks such as monitoring vital signs and performing reminding functions. Joseph Engelberger popularized the concept with the HelpMate robot prototype in the 1990s [53.129], and Mitsubishi Heavy Industries commercialized the Wakamaru robot [53.130] for home use, with clinical trials pending. Even though it is not capable of manipulating objects, it uses its arms to gesture and provide communication cues to people in the home, in addition to using speech output and speech recognition.

53.4 Smart Prostheses and Orthoses

53.4.1 Grand Challenges and Roadblocks

In 2005, the Defense Sciences Office (DSO) of the US governmental research agency, the Defense Advanced Research Projects Agency (DARPA) launched a program to revolutionize prosthetics in a four year timeframe. According to the agency website (<http://www.darpa.mil/dso/thrust/biosci/revprost.htm>), this program will

deliver a prosthetic arm for clinical trials that is far more advanced than any currently available. This device will enable many degrees of freedom for grasping and other hand functions, and will be rugged and resilient to environmental factors. In four years, DSO will deliver a prosthetic for clinical trials that has function almost identical to a natural limb in terms of motor control and dexterity, sensory feedback (including proprioception), weight, and environmental resilience. The four-year device will be directly controlled by neural signals. The results of this program will allow upper limb amputees to have as normal a life as possible despite their severe injuries.

This program announcement lays down the grand challenge for prosthetics research in an ambitious timeframe: develop an artificial limb that has functionality and durability at least as good as a natural limb.

There are several roadblocks to meeting this challenge. First, providing an intuitive way for individuals to control and coordinate multiple joints of a robotic limb is challenging. Second, robots do not yet match the human arm in terms of the combination of range of force, weight, and duration of use with a portable power source. Third, human limbs are rich with tactile and movement sensors. Installing artificial sensors on a robotic limb, and then returning information from those sensors in a way that is usable by the user is challenging. Thus, solving the grand challenge will require better sensory-motor interfaces for prosthetic limbs, as well as lighter stronger actuators and better power sources.

Substantial progress has recently been made in improving sensory-motor interfaces for prosthetic limbs, and this progress is the focus of this section. For the current state of robotic actuators that could be used in prosthetic devices, the reader is referred to Chap. 62 on neurorobotics. For an overview of the design of conventional prosthetic hands and arms, the reader is referred to [53.131].

53.4.2 Targeted Reinnervation

Standard prosthetic arms and hands are commonly controlled with a cable drive or by EMG signals from residual muscles. For example, to open and close an artificial hand, one common technique is to place a Bowden cable around the shoulders in a harness, and connect the cable directly to the artificial hand. The user can then shrug the shoulders to move the cable and open and close the hand. Alternately, electrodes can be placed on a muscle in the residual limb or on the user's back, for example, and then used to control a motor on the artificial hand. The cable technique has the advantages of simplicity, and of having the property of extended physiological proprioception (EPP), which refers to the fact that the grip force is mechanically transmitted back to the user's shoulder muscle force sensors so that the user can gauge the strength of the grasp. Because of their simplicity and EPP, cable drives (or body-powered prostheses) have been more popular than myoelectric (or externally powered) prostheses. However, the body-powered technique is amenable to controlling only one degree of freedom at a time, although chin switches, for example, can be used to switch between degrees of freedom in a somewhat cumbersome way. The myoelectric approach can be used to control multiple degrees of freedom, but such control is nonintuitive and cumbersome. Also, multiple control sites for reading out EMG are not available for people who have lost their entire arm. Thus, prosthetic control systems are typically limited to one or two degrees of freedom, while functional arm and hand movement benefits from at least four degrees of freedom (three to position the hand, and one to open and close it).

Kuiken et al. [53.132] recently developed a novel approach to improving control of a multijointed prosthetic arm. In this targeted reinnervation technique, they rerouted the nerves that previously innervated the lost limb to a spared muscle, and then read out the user's intent to move the limb using electromyography at the spared muscle. They demonstrated this technique in a bilateral shoulder disarticulation amputee who had lost both of his arms in an electrical power accident. They took the residual brachial plexus nerve for the left arm, which normally innervates the left elbow, wrist, and hand, and moved it to the pectoralis muscle. The subject could still contract his pectoralis muscle, but this muscle was no longer useful to him since it used to attach to his now-missing humerus. A surgeon dissected por-

tions of the nerve associated with different muscles in the elbow, wrist, and hand, and innervated three bundles of the pectoralis muscle. After three months, the nerve reinnervated the bundles so that the patient could cause the bundles to twitch by trying to bend his missing elbow, for example. Surface EMG electrodes were placed over the bundles. Then, when the user willed to open his hand, for example, a pectoralis muscle bundle contracted, and this contraction was detectable with the EMG electrodes. The EMG signal was in turn used to control the hand motor of the prosthetic arm. The net result was that the user could will his different (missing) anatomical joints to move, and the corresponding joints on the robotic arm would move. He could simultaneously operate two joints, such as the elbow and the hand. The user became able to do tasks that he was not able to do before with his conventional myoelectric controlled arm, such as feeding himself, shaving, and throwing a ball. A secondary remarkable finding was that the sensory neurons in the rerouted nerves reinnervated sensors, so that now when the person's chest is touched, the person perceives it as a touch to his missing limb. This sensory reinnervation could possibly be made into an interface to provide tactile sensation from the artificial limb. These findings were recently confirmed in another person who received targeted reinnervation [53.133]. It has also recently been shown that direct electrical stimulation of a residual peripheral nerve can provide usable information regarding force to a person with an amputation [53.134].

53.4.3 Brain–Machine Interfaces

There has also recently been progress in decoding movement-related signals in real-time directly from the brain (see the cover story and related articles in *Nature*, Vol. 442 [53.135]). The ability to decode an intended movement directly from brain activity could make it possible to control a prosthetic limb or orthotic device directly by thought.

Brain–computer interfaces can be divided into non-invasive and invasive approaches. In noninvasive approaches, electrical activity is recorded from the surface of the skull using surface electrodes to form an electroencephalogram (EEG) (for a review see [53.136]). Individuals can learn to control the amplitude of the EEG signal as a function of time, or the amplitude of specific frequency components of the EEG signal, with a moderate amount of practice (several hours to several days). The level of control is sufficient to operate a typing program on a computer, or to control the movement of a cursor to multiple targets.

Invasive approaches involve implanting electrodes on the surface of the brain (electrocorticogram) or into the brain itself. Electrodes implanted inside the brain can detect action potentials from single neurons. The first demonstration of this technique in humans [53.137] was in a person with end-stage amyotrophic lateral sclerosis (ALS), a disease that paralyzes muscles but leaves cognition intact. A cone electrode was implanted in the motor cortex, along with growth factors that encouraged growth of neurites, or branches of neurons, into the cone. Stable action potentials were recorded for several months, and the patient was able to increase or decrease the firing rate of the action potentials recorded by the electrode.

Subsequent work in monkeys demonstrated that recordings from multiple neurons (ranging from as few as tens to hundreds) can be used in real time to decode the three-dimensional trajectory of the arm using straightforward signal processing (see review [53.138]). The first human volunteer, a person with tetraplegia due to spinal cord injury, has now been implanted with the BrainGate electrode array, and has been able to control the movement of a cursor on a computer screen [53.135]. Other work in brain–machine interfaces has focused on detecting higher-level control signals, such as the intent to move, preferences for different rewards, and motivation to perform a task [53.139].

Given this progress, it appears that future control systems for smart prosthetics and orthoses will have the option to rely on direct interfaces to the brain, which should allow control of multiple joints through thought alone. The initial work on both targeted reinnervation and brain–machine interfaces has allowed three to four degrees of freedom of control in a naturalistic manner, which is an advance over conventional prosthetic control techniques.

53.4.4 Advances in Neural Stimulation

Unlike people who suffer an amputation, individuals who suffer from paralysis due to neurologic injury retain their limbs. Frustratingly, however, they often do not achieve as good of control as people with artificial limbs. The situation is ironic because the actuator in the limb muscle, is very sophisticated and can still be activated by stimulating the nerves that innervate the muscles. Thus, one viewpoint of the problem of restoring motion for paralyzed people is that excellent hardware for solving the problem already exists (i. e., the limbs themselves), but that we must find ways to replace the control system to control it.

Functional neural stimulation techniques (FNS) seek to stimulate the residual nervous system electrically to reanimate the limbs. FNS for standing, walking, reaching and grasping has been demonstrated, but these techniques have met with limited commercial success because of a combination of factors, including ease of use of systems that use surface electrodes, duration of use before fatigue, risk from implantation and complexity of the associated control problems. The systems that have been most successful are foot-drop stimulators, which aid individuals who suffer from neurologic conditions in lifting a drooping foot while walking. These systems detect the phase of the gait cycle (using a foot switch or accelerometer, for example) and then stimulate a nerve in the leg to lift the foot at the appropriate time.

Two recent innovations may help move the FNS field forward. The first innovation is a hardware innovation. The bionic neuron (BION) is an injectable stimulator the size of a very large grain of cooked rice [53.140]. The initial version is powered through electromagnetic transduction via a coil on the outside of the body, but versions are under development that include a battery. Communication with the individually addressable, programmable stimulator on the BION is through radio frequencies. Sensors, including accelerometers, can also be packaged into the capsule, so that the multiple implanted devices could communicate with themselves to calculate limb position. Two potential advantages of the BION which may make FNS more practical are that it can be inserted without surgery (using a large-gauge needle), and that it may be more robust and resistant to infections, since it does not require implanted wires that can become roadways for infections to spread.

The second innovation that may push FNS forward is stimulating the control circuits in the nervous system rather than individual muscles. For example, it has been shown that locomotor-like movements can be eliciting in multiple muscles of the cat hind limb by stimulating regions of the spinal cord directly (see review [53.141]). The key realization here is that there

exists modular circuitry in the spinal cord, and perhaps elsewhere, that can be electrically activated in order to produce coordinated patterns of muscle activity in multiple muscles. Thus, one electrode can produce coordinated, multijoint muscle activity, reducing the number of needed control sites, and perhaps improving the smoothness and time to fatigue of the resulting contractions because of use of more natural muscle recruitment mechanisms.

53.4.5 Embedded Intelligence

Recent robotics-related advances for prosthetic legs have included embedding microprocessors and passive braking systems into artificial knees, so that the knees can, for example, be made relatively stiff during the stance phase of gait, and free to move during the swing phase of gait (see the review in [53.142]). The first microprocessor knee introduced was the Otto Bock C-Leg, introduced in 1999. The C-Leg uses a servomotor to adjust valves to hydraulic pistons. The rechargeable battery lasts about 24 hours. The pattern of resistance throughout the gait cycle can be adjusted for each user. A dramatic example of the benefit of the C-Leg is the story of a man who made it down from the 70th floor of the World Trade Center on 9th September 2001 with only minor bruising to his residual stump [53.143]. Other microprocessor-controlled knees are the Endolite adaptive prosthesis, which uses pneumatic and hydraulic valves, the Rheo Knee (Ossur hf, Iceland), which uses magnetorheologic fluid to vary the knee impedance, and the Intelligent Prosthesis.

The first powered knee that can generate power, rather than just dissipate energy, is currently being commercialized by Ossur as the Power Knee. The system combines an electromechanical power source that will be controlled with input from sensors on the sound leg shoe. Initial reports suggest that this is the first knee that allows the user to walk up stairs with a step-over-step pattern.

53.5 Augmentation for Diagnosis and Monitoring

53.5.1 Introduction: Grand Challenges and Enabling Technologies

A critical aspect of rehabilitation is health maintenance with age-related or degenerative functional decline and after a medical intervention. In-home diagnostic equipment, devices worn on or in the body for vital signs

monitoring, tele-health services, and institution-based monitoring automation are all examples of systems being developed to improve the quality of life for both persons at risk as well as their care-givers. Institutional systems of this nature, more properly part of the field of clinical engineering, are incorporating more robotic, networked, and autonomous devices to take more accurate

diagnostics, provide better information to physicians, and provide faster alerts. Key enabling technologies in this field are advanced materials and nanotechnologies. For all devices that are worn on the body, the interface must be skin compatible. A grand challenge for this field in the near term is the better incorporation of active and sensing elements with textiles. Several prototype sensor shirts show promise, but rehabilitation will have a much richer toolset for diagnosis and monitoring with advances in this area. Nanotechnology, an enabling technology for the longer term, has the promise to miniaturize virtually everything mechatronic that is currently macroscopic. Injected devices such as nanorobotic drug dispensers and clot-busters will aid in rehabilitation.

53.5.2 Smart Clinics with Automated Health Care Monitoring and Care

A special class of fixed-station rehabilitation robots is an automation system designed to provide a safe environment to assist and monitor a person with a disability living at home or in an institutionalized setting such as an assisted living facility or nursing home.

Smart Nursing Home Automation

An assisted living, hospice or nursing care institutional facility will include residents who have mild to severe cognitive impairment in addition to physical disabilities. The facility may have zones to separate residents who have different levels of dependency since the architectural, monitoring, and personnel needs are different. To better serve residents and guests, to optimize function, and to minimize cost, only the areas for persons with high dependency have a 24 hour staffed vital signs monitoring and alert capability, for example. Facility care is highly staff-intensive, though automation through diagnostic vital signs monitoring, electronic surveillance, and patient tracking continue to improve safety and efficiency. Robotics and automation are beginning to find applications in the physical tasks associated with patient care, therapy, and oversight. Some examples are described below.

Examples of the State of the Art

Wandering, especially at night, is a significant problem for institutional facilities with ambulatory residents who are cognitively impaired. Simple architectural modifications include painting the hallway in front of doors black to make them look like deep holes. An automated voice system triggered by a motion detector to say *Go back to bed* is effective, but not fool-proof, either. Resident

detection systems based on identity badges with embedded radiofrequency identification (RFID) chips that can be sensed in a hallway work, but only if the resident is wearing it. A robotic sentry system, including mobile platforms to aid in solving this problem, especially at night, has not yet been developed.

A serious rehabilitation issue in institutionalized facilities is the transfer of residents from bed to wheelchair and other surfaces. A number of manual, electric, and robotic devices have been developed to assist the nursing staff to safely transfer residents and patients who may be significantly heavier than they are. This remains an unmet clinical need, though not for lack of innovative attempts [53.144, 145].

53.5.3 Home-Based Rehabilitation Monitoring Systems

Numerous smart homes have been developed for non-rehabilitation as well as assistive purposes [53.146]. These systems have as their goal the safety of people with disabilities living in the home and communication with care-givers outside of the home. Care-givers can be live-in family or attendants who, even when they are not home, need to be kept informed on the status of the disabled person, as well as clinicians who need to be sent regular vital signs and other medical/therapy reports. In-home systems typically feature the same principal elements:

1. sensors to monitor both ambient as well as people- and object-specific parameters (e.g., person location, stove-top operation); actuators to modify ambient conditions (heat, lighting, sound system, etc.) and operate devices (doors, refrigerator, etc.);
2. a means to network all the sensors and actuators for uni- or bidirectional communication with the host computer. This network can be wireless (e.g., 802.11g), wired (e.g., coaxial cable), or dependent on an existing network (e.g., signals superimposed on current carried by the electrical mains or phone wiring);
3. a host computer that allows all sensor states to be displayed and actuators to be operated from one or more locations in the home by the inhabitant(s) using common computer input/output (I/O) devices. Higher-order functions are built upon this basic capability;
4. an external network to allow communication with the Internet via phone, cable, satellite or other means. This capability allows for remote monitoring and

operation, sending of alarms and discussions with rehabilitation professionals at medical centers.

The host computer software may also have higher-order features, for example, timers for repetitive actuation of lights and monitoring for anomalous sensor readings (e.g., call security when the smoke detector activates, alert inhabitants with an in-home alarm when stove-top power is on and no pot is on the stove). More advanced features that involve multi-input and multi-output control and adaptive, predictive, context-aware operation [53.147] are areas of active research, and especially important to the rehabilitation community.

53.5.4 Wearable Monitoring Devices

One component of an automated rehabilitation environment is the subsystem that a person wears to be

able to measure, analyze, and communicate physiological signals to an external computer wirelessly. Systems such as the LifeShirt (VivoMetrics, Inc.) [53.148] have been and are being developed for front-line soldiers and rescue operation personnel whose health may become imperiled when out of touch with and unable to communicate verbally with their base command. For rehabilitative purposes, for example the Intel Proactive Health Initiative [53.149] is an example of a system that use on-person position and motion sensing to detect potentially dangerous or undesirable situations.

The most significant obstacles to the widespread adoption of these technologies in the short term are cost, false-positive alarms, inconvenience, and encumbrance. Advances in microelectronics, nanotechnologies, software algorithms, and networking capabilities will drive the research and consumer acceptance of this technology sector.

53.6 Safety, Ethics, Access, and Economics

Rehabilitation robots interact closely with humans, often sharing the same workspace and sometimes physically attaching to humans, as is the case of robotic movement-training devices and prosthetic limbs. Furthermore, the devices are by necessity powerful enough to manipulate the environment or the user's own limbs, which means that they are also often powerful enough to injure the user or another person nearby by colliding with them or moving their limbs inappropriately. Safety is clearly of paramount importance. See Chap. 56 for an in-depth discussion of safety and robotics.

A common strategy for ensuring safety is to incorporate multiple, redundant safety features. A device can be designed to be mechanically incapable of moving itself or the user's limbs in such a way as to cause injury. Limits can be placed on the range, strength, and speed of actuators so that they can accomplish the desired task but no more. Breakaway attachments can be used to attach to users' limbs. Covers can protect the user from pinch points in the device. Redundant sensors can be included, so that if one sensor malfunctions another sensor can identify the malfunction and help shut down the machine safely. Watchdog timers can monitor the health of the control computer. Software checks can limit forces, motions, speeds, and user adjustments to control parameters, as well as check for sensor health and other dangerous situations. Control strategies can be designed so that the device is mechanically compliant,

reducing the risk of forcing a limb into an undesirable configuration, or of a high-impact collision. A manual override switch can be incorporated so that the user can shutdown the system. Finally, the user can be instructed on how to safely operate the device and avoid dangerous situations. Safety ultimately depends, however, on careful and rigorous failure mode analysis and remediation by the system designers.

From a systems perspective, when all else fails actively to protect the user, it must be the design itself that makes the robot inherently unable to injure its user. Part of the solution is in reducing the weight, rounding the surface characteristics and making appropriate materials choices. The goal of inherent safety, however, is often at odds with high performance and adequate payload for real-life tasks. Recently, several approaches to designing personal robots – in other words the class to which assistive rehabilitation robots belong – have sought to address both goals by dividing the two tasks of compensating for gravity (arm plus payload) and moving the payload around in space [53.150]. The solution is to provide two actuators per joint on the joints that support the arm segments and payload against gravity: one slow, gear-reduced motor and energy-storing device such as a large spring or compressed air volume, and one small, back-drivable motor that provides the power needed to move objects around quickly and precisely. Most robot manipulators have approximately a 1:10 (or worse) payload-to-weight ratio. A system with a dual,

parallel actuator system that requires only the small, fast actuator to be carried in the arm, leaving the slow energy-storage system on the base and not contributing to the inertia of the arm itself, can lead to a 1:1 payload-to-weight ratio that is more in line with a human's own arm characteristics, and thereby provides a safe yet high-performance solution. An added benefit of this type of arrangement is that the movements of the arm will tend to be more human-like, providing a measure of confidence to the user that the robot is performing properly and moving in a safe way.

Strategies for improving safety have been proposed and methods to assess safety have been developed and adapted for rehabilitation robotics [53.151, 152] based on accepted risk analysis methods. While industrial robots have benefited from International Organization for Standardization (ISO) user safety regulations since 1992 (ISO 10218), the fundamental issue of human proximity to robots for the personal, service, and rehabilitation sectors have prevented any consensus to date for a similar standard. Currently, the existing industrial standards, augmented with provisions from medical equipment standards and buttressed by engineering best practices and adherence to professional codes of ethics by designers, have guided rehabilitation robotics designers. Clearly, as products appear on the market and the expected rapid expansion of this sector happens, better regulations must be developed.

Beyond safety, there are other ethical concerns that will emerge as robotics technology becomes more intelligent with advances in cognitive software, more invasive with nanotechnologies, and better integrated with human systems through bioengineering advances. Ethicists and roboticists are starting to deal with these issues [53.153, 154], which to date have been the purview of only futurists and science fiction writers. Chapter 64 in this Handbook deals with these issues in detail.

An economic advantage has not yet been demonstrated in a decisive way for most rehabilitation robotics. For example, the therapeutic benefits conferred by robotic therapy devices, and the assistive benefits conferred by wheelchair-mounted robots relative to the device cost, have not yet been large enough to cause widespread adoption. Improvements in their efficacy and reductions in their cost will increase their usage. For example, a robotic therapy device that helps people learn to walk after a stroke in a way that is decisively better than other training techniques would become widely used very quickly. Likewise, a wheelchair-mounted robot that gives a disabled person a substantial and efficient increase in autonomy at a reasonable cost would also quickly become widely used. An example of a robotics technology that has achieved an attractive cost-benefit ratio and thus is commercially successful is the powered wheelchair.

53.7 Conclusions and Further Readings

Rehabilitation robotics is a dynamic application area because its grand challenges are at the forefront of both robotics and biology research. The ongoing major themes of the field can be summarized as the development of robotic therapy devices, smart prostheses, orthoses, functional aids, and nurses that match or exceed the capabilities of their human counterparts. Rehabilitation robotics is also a highly motivating field because the technology developed will directly help people who are limited in major life activities. The field will continue to grow because of the dramatic aging of the populations of industrialized countries that is just beginning.

The grand challenges of rehabilitation robotics are grounded in the distinguishing features of the field: functional involvement with humans, a physical user interface, and behavior that is intelligent, adaptive, and safe. These characteristics require high levels of redundancy, sensorimotor capability, adaptability, and

multilevel software architecture. The grand challenges therefore span the domains of electromechanical design, software design, and, due to the applied and innately human-focused nature of rehabilitation robotics, all aspects of user interface design, including physical, communication, learning, emotional, and motivational factors. The first products in this field have come on the market in only the last ten years; worldwide demographic trends will provide the force to accelerate product development in the future.

For further investigation on rehabilitation robotics, there are three major sources of published information: (1) books on personal, service, and rehabilitation robotics such as [53.155–157]; (2) review articles in journals and periodicals such as [53.158–161]; and (3) articles that deal with individual topics, such as those in the reference list below, and conferences such as the International Conference on Rehabilitation

Robotics (ICORR), Rehabilitation Engineering and Assistive Technology Society of North America (RESNA) conferences, RO-MAN, and the Association for the Advancement of Assistive Technology in Europe (AAATE) conferences, which are also represented in the refer-

ence list. Cutting-edge research will be reported on the websites of investigators at academic, government, and corporate research labs, and it is recommended to start at the sites of the researchers cited in this chapter.

References

- 53.1 L. Leifer: Rehabilitative robots. In: *Robotics Age: In the Beginning selected from Robotics Age Magazine*, ed. by C. Helmers (Hayden, Hasbrouck 1981) pp. 227–241
- 53.2 M. Kassler: Introduction to the special issue on robotics for health care, *Robotica* **11**, 493–494 (1993)
- 53.3 H.F.M. Van der Loos: VA/Stanford rehabilitation robotics research and development program: lessons learned in the application of robotics technology to the field of rehabilitation, *IEEE Trans. Rehab. Eng.* **3**, 46–55 (1995)
- 53.4 J.L. Patton, M.E. Phillips-Stoykov, M. Stojakovich, F.A. Mussa-Ivaldi: Evaluation of robotic training forces that either enhance or reduce error in chronic hemiparetic stroke survivors, *Exp. Brain Res.* **168**, 368–383 (2006)
- 53.5 J. Emken, D. Reinkensmeyer: Robot-enhanced motor learning: Accelerating internal model formation during locomotion by transient dynamic amplification, *IEEE Trans. Neural Syst. Rehab. Eng.* **99**, 1–7 (2005)
- 53.6 H.F.M. Van der Loos, R. Mahoney, C. Ammi: Great expectations for rehabilitation mechatronics in the coming decade. In: *Advances in Rehabilitation Robotics*, Lect. Notes Contr. Inf. Sci. **306**, 427–433 (2004)
- 53.7 K. Corker, J.H. Lyman, S. Sheredos: A preliminary evaluation of remote medical manipulators, *Bull. Prosth. Res.* **10**, 107–134 (1979)
- 53.8 M. Hillman: Rehabilitation robotics from past to present – A historical perspective. In: *Advances in Rehabilitation Robotics*, Lect. Notes Contr. Inf. Sci. **306**, 25–44 (2004)
- 53.9 W. Seamone, G. Schmeisser: Early clinical evaluation of a robot arm/work table system for spinal cord injured persons, *J. Rehab. Res. Dev.* **22**, 38–57 (1985)
- 53.10 J. Guittet, H.H. Kwee, N. Quetin, J. Yelon: The SPAR-TACUS telethesis: manipulator control studies, *Bull. Prosth. Res.* **10**, 69–105 (1979)
- 53.11 J. Hammel, K. Hall, D.S. Lees, L.J. Leifer, H.F.M. Van der Loos, I. Perkash, R. Crigler: Clinical evaluation of a desktop robotic assistant, *J. Rehab. Res. Dev.* **26**, 1–16 (1989)
- 53.12 J.M. Hammel, H.F.M. Van der Loos, I. Perkash: Evaluation of a vocational robot with a quadriplegic employee, *Arch. Phys. Med. Rehabil.* **73**, 683–693 (1992)
- 53.13 H.F.M. Van der Loos, S.J. Michalowski, L.J. Leifer: Design of an omnidirectional mobile robot as a manipulation aid for the severely disabled. In: *Interactive Robotic Aids*, ed. by R. Foulds (World Rehabilitation Fund, New York 1986) pp. 61–63
- 53.14 J.J. Wagner, M. Wickizer, H.F.M. Van der Loos, L.J. Leifer: User testing and design iteration of the ProVAR user interface, presented at 8th IEEE International Workshop on Robot and Human Interaction: RO-MAN'99 (1999)
- 53.15 M. Busnel, R. Cammoun, F. Coulon-Lauture, J.-M. Detriche, G. Le Claire, B. Lesigne: The robotized workstation MASTER for users with tetraplegia: Description and evaluation, *J. Rehab. Res. Dev.* **36**, 217–229 (1999)
- 53.16 R. Gelin, B. Lesigne, M. Busnel, J.P. Michel: The first moves of the AFMASTER workstation, *Adv. Robot.* **14**, 639–649 (2001)
- 53.17 M. Topping: The development of Handy 1, a rehabilitation robotic system to assist the severely disabled, *Ind. Robot.* **25**, 316–320 (1998)
- 53.18 R.M. Mahoney: The Raptor wheelchair robot system. In: *Integration of Assistive Technology in the Information Age* (IOS, Amsterdam 2001) pp. 135–141
- 53.19 H.H. Kwee: Integrated control of MANUS manipulator and wheelchair enhanced by environmental docking, *Robotica* **16**, 491–498 (2000)
- 53.20 Z. Bien, M.J. Chung, P.H. Chang, D.S. Kwon, D.J. Kim, J.S. Han, J.H. Kim, D.H. Kim, H.S. Park, S.H. Kang: Integration of a rehabilitation robotic system (KARES II) with human-friendly man-machine interaction units, *Auton. Robot.* **16**, 165–191 (2004)
- 53.21 R.C. Simpson, S.P. Levine, D.A. Bell, L. Jaros, Y. Koren, J. Borenstein: NavChair: An assistive wheelchair navigation system with automatic adaptation. In: *Assistive Technology and AI*, Lect. Notes Artif. Intell. **1458**, 235–255 (1998)
- 53.22 R.C. Simpson, D. Poirot, F. Baxter: The Hephaestus smart wheelchair system, *IEEE Trans. Neural Syst. Rehab. Eng.* **10**, 118–122 (2002)
- 53.23 R. Krukowski: BioDex Muscle exercise and rehabilitation apparatus, US Patent No. 4765315 (Int. A63B 2300) (1986)

- 53.24 D. Khalili, M. Zomlefer: An intelligent robotic system for rehabilitation of joints and estimation of body segment parameters, *IEEE Trans. Biomed. Eng.* **35**, 138–146 (1988)
- 53.25 M.P. Dijkers, P.C. deBear, R.F. Erlandson, K.A. Kristy, D.M. Geer, A. Nichols: Patient and staff acceptance of robotic technology in occupational therapy: a pilot study, *J. Rehab. Res.* **28**, 33–44 (1991)
- 53.26 K. Dautenhahn, I. Werry: Towards interaction robots in autism therapy: background, motivation, and challenges, *Pragmact. Cognit.* **12**, 1–35 (2004)
- 53.27 T. Shibata, T. Mitsui, K. Wada, K. Tanie: Psychophysiological effects by interaction with mental commit robot, *J. Robot. Mechatron.* **14**, 13–19 (2002)
- 53.28 L. Sawaki: Use-dependent plasticity of the human motor cortex in health and disease, *IEEE Eng. Med. Biol. Mag.* **24**, 36–39 (2005)
- 53.29 J.R. Wolpaw, A.M. Tennissen: Activity-dependent spinal cord plasticity in health and disease, *Annu. Rev. Neurosci.* **24**, 807–843 (2001)
- 53.30 K.M. Baldwin, F. Haddad: Skeletal muscle plasticity: cellular and molecular responses to altered physical activity paradigms, *Am. J. Phys. Med. Rehabil.* **81**, S40–S51 (2002)
- 53.31 J.L. Emken, R. Benitez, D.J. Reinkensmeyer: Human-robot cooperative movement training: learning a novel sensory motor transformation during walking with robotic assistance-as-needed, *J. Neuroeng. Rehab.* **4**, 8 (2007)
- 53.32 B.H. Dobkin: *Neurologic Rehabilitation* (F. A. Davis, Philadelphia 1996)
- 53.33 H.I. Krebs, N. Hogan, M.L. Aisen, B.T. Volpe: Robot-aided neurorehabilitation, *IEEE Trans. Rehabil. Eng.* **6**, 75–87 (1998)
- 53.34 S.P. Buerger, H.I. Krebs, N. Hogan: Characterization and control of a screw-driven robot for neurorehabilitation, *Proceedings of the 2001 IEEE International Conference on Control Applications* pp. 388–394 (2001)
- 53.35 D.J. Williams, H.I. Krebs, N. Hogan: A robot for wrist rehabilitation, *Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society* **2**, (2001) pp. 1336–1339
- 53.36 L. Masia, H.I. Krebs, P. Cappa, N. Hogan: Whole-arm rehabilitation following stroke: Hand module, *BioRob 2006. The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechanics: February 20–22* (2006) pp. 1085–1089
- 53.37 J. Stein, H.I. Krebs, W.R. Frontera, S. Fasoli, R. Hughes, N. Hogan: Comparison of two techniques of robot-aided upper limb exercise training after stroke, *Am. J. Phys. Med. Rehabil.* **83**, 720–728 (2004)
- 53.38 H. Krebs, J. Palazzolo, L. Dipietro, M. Ferraro, J. Krol, K. Rannekleiv, B. Volpe, N. Hogan: Rehabilitation robotics: performance-based progressive robot-assisted therapy, *Auton. Robot.* **15**, 7–20 (2003)
- 53.39 M.L. Aisen, H.I. Krebs, N. Hogan, F. McDowell, B. Volpe: The effect of robot-assisted therapy and rehabilitative training on motor recovery following stroke, *Arch. Neurol.* **54**, 443–446 (1997)
- 53.40 S. Fasoli, H. Krebs, J. Stein, W. Frontera, N. Hogan: Effects of robotic therapy on motor impairment and recovery in chronic stroke, *Arch. Phys. Med. Rehabil.* **84**, 477–482 (2003)
- 53.41 J.J. Daly, N. Hogan, E.M. Perepezko, H.I. Krebs, J.M. Rogers, K.S. Goyal, M.E. Dohring, E. Fredrickson, J. Nethery, R.L. Ruff: Response to upper-limb robotics and functional neuromuscular stimulation following stroke, *J. Rehabil. Res. Dev.* **42**, 723–736 (2005)
- 53.42 P.S. Lum, C.G. Burgar, P.C. Shor, M. Majmundar, H.F.M. Van der Loos: Robot-assisted movement training compared with conventional therapy techniques for the rehabilitation of upper limb motor function following stroke, *Arch. Phys. Med. Rehabil.* **83**, 952–959 (2002)
- 53.43 P.S. Lum, C.G. Burgar, H.F.M. Van der Loos, P.C. Shor, M. Majmundar, R. Yap: MIME robotic device for upper-limb neurorehabilitation in sub-acute stroke subjects: A follow-up study, *J. Rehab. Res. Dev.* **43**, 631–642 (2006)
- 53.44 L.E. Kahn, M.L. Zygmant, W.Z. Rymer, D.J. Reinkensmeyer: Robot-assisted reaching exercise promotes arm movement recovery in chronic hemiparetic stroke: A randomized controlled pilot study, *J. Neuroeng. Neurorehab.* **3**, 12 (2006)
- 53.45 P.S. Lum, D.J. Reinkensmeyer, S.L. Lehman: Robotic assist devices for bimanual physical therapy: preliminary experiments, *IEEE Trans. Rehabil. Eng.* **1**, 185–191 (1993)
- 53.46 S. Hesse, C. Werner, M. Pohl, S. Rueckriem, J. Mehrholz, M.L. Lingnau: Computerized arm training improves the motor control of the severely affected arm after stroke: a single-blinded randomized trial in two centers, *Stroke* **36**, 1960–1966 (2005)
- 53.47 D. Reinkensmeyer, J. Emken, S. Cramer: Robotics, motor learning, and neurologic recovery, *Ann. Rev. Biomed. Eng.* **6**, 497–525 (2004)
- 53.48 F. Amirabdollahian, E. Gradwell, R. Loureiro, W. Harwin: Effects of the GENTLE/S robot mediated therapy on the outcome of upper limb rehabilitation post-stroke: Analysis of the Battle Hospital data, *Proceedings of the 8th International Conference on Rehabilitation Robotics, Daejeon, Korea* (2003) pp. 55–58
- 53.49 F. Amirabdollahian, R. Loureiro, E. Gradwell, C. Collin, W. Harwin, G. Johnson: Multivariate analysis of the Fugl-Meyer outcome measures assessing the effectiveness of GENTLE/S robot-mediated stroke therapy, *J. Neuroeng. Rehab.* **19**, 4 (2007)

- 53.50 A.S. Merians, H. Poizner, R. Boian, G. Burdea, S. Adamovich, M. Kuttuva, A. Merians, M. Bouzit, J. Lewis, D. Fensterheim: Sensorimotor training in a virtual reality environment: does it improve functional recovery poststroke? The Rutgers Arm, a rehabilitation system in virtual reality: a pilot study, *Neurorehab. Neural Repair* **20**, 252–267 (2006)
- 53.51 R. Colombo, F. Pisano, S. Micera, A. Mazzone, C. Delconte, M. Carrozza, P. Dario, G. Minuco: Robotic techniques for upper limb evaluation and rehabilitation of stroke patients, *IEEE Trans. Neural Syst. Rehabil. Eng.* **13**, 311–324 (2005)
- 53.52 R.J. Sanchez, J. Liu, S. Rao, P. Shah, R. Smith, S.C. Cramer, J.E. Bobrow, D.J. Reinkensmeyer: Automating arm movement training following severe stroke: functional exercises with quantitative feedback in a gravity-reduced environment, *IEEE Trans. Neural Syst. Rehabil. Eng.* **14**, 378–389 (2006)
- 53.53 S. Masiero, A. Celia, G. Rosati, M. Armani: Robotic-assisted rehabilitation of the the upper limb after acute stroke, *Arch. Phys. Med. Rehabil.* **88**, 142–149 (2007)
- 53.54 G. Fazekas, M. Horvath, A. Toth: A novel robot training system designed to supplement upper limb physiotherapy of patients with spastic hemiparesis, *Int. J. Rehabil. Res.* **29**, 251–254 (2006)
- 53.55 T. Nef, R. Riener: ARMin – Design of a novel arm rehabilitation robot, *Proceedings of the 2005 IEEE International Conference on Rehabilitation Robotics, June 28–July 1, Chicago, Illinois* (2005) pp. 57–60
- 53.56 E. Wolbrecht, J. Leavitt, D. Reinkensmeyer, J. Bobrow: Control of a pneumatic orthosis for upper extremity stroke rehabilitation, *IEEE Engineering in Medicine and Biology Conference, New York* (2006) pp. 2687 – 2693
- 53.57 J.L. Patton, G. Dawe, C. Scharver, F.A. Mussa-Ivaldi, R. Kenyon: Robotics and virtual reality: the development of a life-sized 3-D system for the rehabilitation of motor function, *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society* (2004) pp. 4840–4843
- 53.58 H. Huang, J. He: Utilization of biomechanical modeling in design of robotic arm for rehabilitation of stroke patients, *26th Ann. Int. Conf. IEEE Eng. Med. Biol. Soc.* **4**, 2718–2721 (2004)
- 53.59 D. Reinkensmeyer, C. Pang, J. Nessler, C. Painter: Web-based telerehabilitation for the upper-extremity after stroke, *IEEE Trans. Neural Sci. Rehabil. Eng.* **10**, 1–7 (2002)
- 53.60 M.J. Johnson, H.F.M. Van der Loos, C.G. Burgar, P. Shor, L.J. Leifer: Experimental results using force-feedback cueing in robot-assisted stroke therapy, *IEEE Trans. Neural Syst. Rehabil. Eng.* **13**, 335–348 (2005)
- 53.61 X. Feng, J. Winters: UniTherapy: a computer-assisted motivating neurorehabilitation platform for teleassessment and remote therapy, *2005 IEEE 9th International Conference on Rehabilitation Robotics* (2005) pp. 349–352
- 53.62 H. Sugarman, E. Dayan, A. Weisel-Eichler, J. Tiran: The Jerusalem TeleRehabilitation System, a new low-cost, haptic rehabilitation approach, *Cyberpsychol. Behav.* **9**, 178–182 (2006)
- 53.63 M. Mulas, M. Folgheraiter, G. Gini: An EMG-controlled exoskeleton for hand rehabilitation, *Proceedings of the 2005 IEEE International Conference on Rehabilitation Robotics, June 28–July 1, Chicago, Illinois* (2005) pp. 371–374
- 53.64 C. Takahashi, L. Der-Yeghiaian, V.H. Le, S.C. Cramer: A robotic device for hand motor therapy after stroke, *Proceedings of the 2005 IEEE International Conference on Rehabilitation Robotics, June 28–July 1, Chicago, Illinois* (2005) pp. 17–20
- 53.65 T. Kline, D.G. Kamper, B.D. Schmit: Control system for pneumatically controlled glove to assist in grasp activities, *Proceedings of the 2005 IEEE International Conference on Rehabilitation Robotics, June 28–July 1, Chicago, Illinois* (2005) pp. 78–81
- 53.66 B.R. Brewer, M. Fagan, R.L. Klatzky, Y. Matsuoaka: Perceptual limits for a robotic rehabilitation environment using visual feedback distortion, *IEEE Trans. Neural Syst. Rehabil. Eng.* **13**, 1–11 (2005)
- 53.67 J.L. Patton, M. Kovic, F.A. Mussa-Ivaldi: Custom-designed haptic training for restoring reaching ability to individuals with poststroke hemiparesis, *J. Rehabil. Res. Dev.* **43**, 643–656 (2006)
- 53.68 R.G. Lovely, R.J. Gregor, R.R. Roy, V.R. Edgerton: Effects of training on the recovery of full weight-bearing stepping in the adult spinal cat, *Exp. Neurol.* **92**, 421–435 (1986)
- 53.69 H. Barbeau, S. Rossignol: Recovery of locomotion after chronic spinalization in the adult cat, *Brain Res.* **412**, 84–95 (1987)
- 53.70 M. Visintin, H. Barbeau, N. Korner-Bitensky, N. Mayo: A new approach to retrain gait in stroke patients through body weight support and treadmill stimulation, *Stroke* **29**, 1122–1128 (1998)
- 53.71 S. Hesse, C. Bertelt, M. Jahnke, A. Schaffrin, P. Baake, M. Malezic, K. Mauritz: Treadmill training with partial body weight support compared with physiotherapy in nonambulatory hemiparetic patients, *Stroke* **26**, 976–981 (1995)
- 53.72 A. Wernig, A. Nanassy, S. Muller: Maintenance of locomotor abilities following Laufband (treadmill) therapy in para- and tetraplegic persons: follow-up studies, *Spinal Cord* **36**, 744–749 (1998)
- 53.73 A.L. Behrman, S.J. Harkema: Locomotor training after human spinal cord injury: a series of case studies, *Phys. Therapy* **80**, 688–700 (2000)
- 53.74 H. Barbeau: Locomotor training in neurorehabilitation: emerging rehabilitation concepts, *Neurorehab. Neural Repair* **17**, 3–11 (2003)

- 53.75 B. Dobkin, D. Apple, H. Barbeau, M. Basso, A. Behrman, D. Deforge, J. Ditunno, G. Dudley, R. Elashoff, L. Fugate: Weight-supported treadmill vs over-ground training for walking after acute incomplete SCI, *Neurology* **66**, 484–493 (2006)
- 53.76 T.G. Hornby: Clinical and quantitative evaluation of robotic-assisted treadmill walking to retrain ambulation after spinal cord injury, *Topics Spinal Cord Injury Rehab.* **11**, 1–17 (2005)
- 53.77 L. Nilsson, K. Fugl-Meyer, L. Kristensen, B. Sjölund, K. Sunnerhagen: Walking training of patients with hemiparesis at an early stage after stroke: a comparison of walking training on a treadmill with body weight support and walking training on the ground, *Clin. Rehab.* **15**, 515–527 (2001)
- 53.78 C. Werner, A. Bardeleben, K. Mauritz, S. Kirker, S. Hesse: Treadmill training with partial body weight support and physiotherapy in stroke patients: a preliminary comparison, *Eur. J. Neurol.* **9**, 639–644 (2002)
- 53.79 S. Hesse, C. Werner, H. Seibel, S. von Frankenberg, E. Kappel, S. Kirker, M. Kading: Treadmill training with partial body-weight support after total hip arthroplasty: A randomized controlled trial, *Arch. Phys. Med. Rehabil.* **84**, 1767–1773 (2003)
- 53.80 T. Brown, J. Mount, B. Rouland, K. Kautz, R. Barnes, J. Kim: Body weight-supported treadmill training versus conventional gait training for people with chronic traumatic brain injury, *J. Head Trauma Rehabil.* **20**, 402–415 (2005)
- 53.81 S. Hesse, D. Uhlenbrock: A mechanized gait trainer for restoration of gait, *J. Rehab. Res. Dev.* **37**, 701–708 (2000)
- 53.82 G. Colombo, M. Joerg, R. Schreier, V. Dietz: Treadmill training of paraplegic patients with a robotic orthosis, *J. Rehab. Res. Dev.* **37**, 693–700 (2000)
- 53.83 HealthSouth: <http://www.autoambulator.com> (2007)
- 53.84 C. Werner, S. Von Frankenberg, T. Treig, M. Konrad, S. Hesse: Treadmill training with partial body weight support and an electromechanical gait trainer for restoration of gait in subacute stroke patients: a randomized crossover study, *Stroke* **33**, 2895–2901 (2002)
- 53.85 M. Pohl, C. Werner, M. Holzgraefe, G. Krocze, J. Mehrholz, I. Wingendorf, G. Hoolig, R. Koch, S. Hesse: Repetitive locomotor training and physiotherapy improve walking and basic activities of daily living after stroke: a single-blind, randomized multicentre trial (DEutsche GAngtrainerStudie, DEGAS), *Clin. Rehabil.* **21**, 17–27 (2007)
- 53.86 M. Wirz, D.H. Zemon, R. Rupp, A. Scheel, G. Colombo, V. Dietz, T.G. Hornby: Effectiveness of automated locomotor training in patients with chronic incomplete spinal cord injury: A multicenter trial, *Arch. Phys. Med. Rehabil.* **86**, 672–680 (2005)
- 53.87 H. Schmidt, S. Hesse, R. Bernhardt, J. Krüger: HapticWalker – a novel haptic foot device, *ACM Trans. Appl. Percept. (TAP)* **2**, 166–180 (2005)
- 53.88 S. Jezernik, G. Colombo, M. Morari: Automatic gait-pattern adaptation algorithms for rehabilitation with a 4-DOF robotic orthosis, *IEEE Trans. Robot. Automat.* **20**, 574–582 (2004)
- 53.89 D. Reinkensmeyer, D. Aoyagi, J. Emken, J. Galvez, W. Ichinose, G. Kerdanyan, S. Maneekobkunwong, K. Minakata, J. Nessler, W. Timoszyk, K. Vallance, R. Weber, J. Wynne, R. Roy, R. d. Leon, J. Bobrow, S. Harkema, V. Edgerton: Tools for understanding and optimizing robotic gait training, *J. Rehab. Res. Dev.* **43**, 657–670 (2006)
- 53.90 D. Surdilovic, R. Bernhardt: STRING-MAN: A new wire robot for gait rehabilitation, presented at *ICRA '04. 2004 IEEE International Conference on Robotics and Automation* (2004)
- 53.91 J.F. Veneman, R. Ekkelenkamp, R. Kruidhof, F.C.T. van der Helm, H. van der Kooij: A series elastic- and bowden-cable-based actuation system for use as torque actuator in exoskeleton-type robots, *Int. J. Robot. Res.* **25**, 261–282 (2006)
- 53.92 J.A. Galvez, D.J. Reinkensmeyer: Robotics for gait training after spinal cord injury, *Topics Spinal Cord Injury Rehab.* **11**, 18–33 (2005)
- 53.93 D. Aoyagi, W.E. Ichinose, S.J. Harkema, D.J. Reinkensmeyer, J.E. Bobrow: An assistive robotic device that can synchronize to the pelvic motion during human gait training, *Proc. 2005 IEEE International Conference on Rehabilitation Robotics, Chicago, Illinois* (2005) pp. 565–568
- 53.94 S. Jezernik, R. Scharer, G. Colombo, M. Morari: Adaptive robotic rehabilitation of locomotion: A clinical study in spinally injured individuals, *Spinal Cord* **41**, 657–666 (2003)
- 53.95 D. Aoyagi, Doctoral Dissertation, *Department of Mechanical and Aerospace Engineering, University of California at Irvine* (2006)
- 53.96 G. Pratt, M. Williamson, P. Dillworth, J. Pratt, K. Ulland, A. Wright: Stiffness isn't everything, *Fourth International Symposium on Experimental Robotics* (1995)
- 53.97 D. Robinson: Design and Analysis of Series Elasticity in Closed-loop Actuator Force Control (Massachusetts Institute of Technology 2000)
- 53.98 D.W. Robinson, J.E. Pratt, D.J. Paluska, G.A. Pratt: Series elastic actuator development for a biomimetic walking robot, 1999 IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (1999) pp. 561–568
- 53.99 M. Peshkin, D.A. Brown, J.J. Santos-Munne, A. Makhlin, E. Lewis, J.E. Colgate, J. Patton, D. Schwandt: KineAssist: A robotic overground gait and balance training device, *ICORR: 9th International Conference on Rehabilitation Robotics* (2005) pp. 241–246

- 53.100 D. Ferris: Powered lower limb orthoses for gait rehabilitation, *Topics Spinal Cord Injury Rehab.* **11**, 34–49 (2005)
- 53.101 J. Deutsch, J. Latonio, G. Burdea, R. Boian: Post-stroke rehabilitation with the Rutgers Ankle system – a case study, *Presence* **10**, 416–430 (2001)
- 53.102 S. Agrawal, A. Fattah: Theory and design of an orthotic device for full or partial gravity-balancing of a human leg during motion, *IEEE Trans. Neural Syst. Rehab. Eng.* **12**, 157–165 (2004)
- 53.103 S.K. Banala, S.K. Agrawal: Gait rehabilitation with an active leg orthosis, *Proceedings of ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Long Beach, CA, USA* (2005)
- 53.104 J. Kawamura, T. Ide, S. Hayashi, H. Ono, T. Honda: Automatic suspension device for gait training, *Prosthet. Orthot. Int.* **17**, 120–125 (1993)
- 53.105 R.F. Erlandson, P. deBear, K. Kristy, M. Dijkers, S. Wu: A robotic system to provide movement therapy, *Proc. 5th Int. Service Robot Conf., Detroit, MI* (1990) pp. 7–15
- 53.106 M.J. Rosen: Telerehabilitation, *NeuroRehabilitation* **12**, 11–26 (1999)
- 53.107 C. Carignan, H. Krebs: Telerehabilitation robotics: Bright lights, big future?, *J. Rehabil. Res. Dev.* **43**, 695–710 (2006)
- 53.108 C. Stanger, M. Cawley: Demographics of rehabilitation robotics users, *Technol. Disability* **5**, 125–138 (1996)
- 53.109 J. Hammel: The role of assessment and evaluation in rehabilitation robotics research and development: Moving from concept to clinic to context, *IEEE Trans. Rehab. Eng.* **3**, 56–61 (1995)
- 53.110 J.J. Wagner, H.F.M. Van der Loos, L.J. Leifer: Dual-character based user interface design for an assistive robot, *ROMAN-98 Conference* (1998) pp. 101–106
- 53.111 H.F.M. Van der Loos, J. Hammel, D.S. Lees, D. Chang, I. Perkas: Field evaluation of a robot workstation for quadriplegic office workers, *Eur. Rev. Biomed. Tech.* **5**, 317–319 (1990)
- 53.112 J.J. Wagner, H.F.M. Van der Loos, L.J. Leifer: Construction of social relationships between user and robot, *Robot. Autonom. Syst.* **31**, 185–191 (2000)
- 53.113 M.J. Johnson, E. Guglielmelli, G.A. Di Lauro, C. Laschi, M.C. Carrozza, P. Dario: GIVING-A-HAND System: The development of a task-specific robot appliance. In: *Advances in Rehabilitation Robotics*, Vol. 306, ed. by Z.Z. Bien, D. Stefanov (Springer, Berlin 2004) pp. 127–141
- 53.114 H.F.M. Van der Loos, N. Ullrich, H. Kobayashi: Development of Sensate and Robotic Bed Technologies for Vital Signs Monitoring and Sleep Quality Improvement, *Auton. Robot.* **15**, 67–79 (2003)
- 53.115 T. Sato, T. Harada, T. Mori: Environment-type robot system Robotic Room featured by behavior media, behavior contents, and behavior adaptation, *IEEE/ASME Trans. Mechatron.* **9**, 529–534 (2004)
- 53.116 G. Romer, H.J.A. Stuyt, A. Peters: Cost-savings and economic benefits due to the Assistive Robotic Manipulator (ARM), *9th International Conference on Rehabilitation Robotics: ICORR* (2005) pp. 201–204
- 53.117 J.F. Engelberger: Health-care robotics goes commercial: The HelpMate experience, *Robotica* **11**, 517–524 (1993)
- 53.118 P. Dario, C. Laschi, E. Guglielmelli: Design and experiments on a personal robotic assistant, *Adv. Robot.* **13**, 153–169 (1999)
- 53.119 B. Graf, M. Hans, R.D. Schraft: Care-O-bot II – Development of a next generation robotic home assistant, *Auton. Robot.* **16**, 193–205 (2004)
- 53.120 S.P. Levine, D.A. Bell, L.A. Jaros, R.C. Simpson, Y. Koren, J. Borenstein: The NavChair assistive wheelchair navigation system, *Rehabilitation Engineering, IEEE Trans.* **7**, 443–451 (1999), [see also *IEEE Trans. on Neural Systems and Rehabilitation*]
- 53.121 H. Yanco: Wheellesley: A robotic wheelchair system: Indoor navigation and user interface, *Assist. Technol. Artif. Intell.* (1998) pp. 256–268
- 53.122 G. Lacey, S. MacNamara: User involvement in the design and evaluation of a smart mobility aid, *J. Rehab. Res. Dev.* **37**, 6 (2000)
- 53.123 K. Wada, T. Shibata, T. Saito, K. Tanie: Effects of three months robot assisted activity to depression of elderly people who stay at a health service facility for the aged., *SICE Annual Conference, Sapporo, Japan* (2004) pp. 2004–2714
- 53.124 A.J. Brisben, A.D. Lockerd, C. Lathan: Design evolution of an interactive robot for therapy, *Telemed. J. E-Health* **10**, 252–259 (2004)
- 53.125 Y. Kusuda: How Japan sees the robotics for the future: observation at the World Expo 2005, *Ind. Robot.* **33**, 11–18 (2006)
- 53.126 J. Eriksson, M. Mataric, C. Winstein: Hands-off assistive robotics for post-stroke arm rehabilitation, *Proceedings of the 2005 IEEE International Conference on Rehabilitation Robotics, June 28–July 1, Chicago, Illinois* (2005) pp. 21–24
- 53.127 M.K. Holden: Virtual environments for motor rehabilitation: review, *Cyberpsychol. Behav.* **8**, 187–211 (2005), discussion 212–219
- 53.128 B. Robins, K. Dautenhahn, R. Te Boekhorst, A. Billard: Robotic assistants in therapy and education of children with autism: can a small humanoid robot help encourage social interaction skills, *Univ. Access Inf. Soc.* **4**, 105–120 (2005)
- 53.129 G. Engelberger: HelpMate, a service robot with experience, *Ind. Robot.* **25**, 101–104 (1998)
- 53.130 M.H. Industries: Life with a Robot: Wakamaru (2003)
- 53.131 R.F. f. Weir: Design of artificial arms and hands for prosthetic applications. In: *Standard handbook of biomedical engineering and design*, ed. by M. Kutz (McGraw-Hill, New York 2003) pp. 32.1–32.61

- 53.132 T.A. Kuiken, G.A. Dumanian, R.D. Lipschutz, L.A. Miller, K.A. Stubblefield: The use of targeted muscle reinnervation for improved myoelectric prosthesis control in a bilateral shoulder disarticulation amputee, *Prosthet. Orthotics Int.* **28**(3), 245–253 (2004)
- 53.133 T.A. Kuiken, L.A. Miller, R.D. Lipschutz, B.A. Lock, K. Stubblefield, P.D. Marasco, P. Zhou, G.A. Dumanian: Targeted reinnervation for enhanced prosthetic arm function in a woman with a proximal amputation: a case study, *Lancet* **369**, 371–380 (2007)
- 53.134 G.S. Dhillon, K.W. Horch: Direct neural sensory feedback and control of a prosthetic arm, *IEEE Trans. Neural Syst. Rehabil. Eng.* **13**, 468–472 (2005)
- 53.135 L.R. Hochberg, M.D. Serruya, G.M. Friehe, J.A. Mukand, M. Saleh, A.H. Caplan, A. Branner, D. Chen, R.D. Penn, J.P. Donoghue: Neuronal ensemble control of prosthetic devices by a human with tetraplegia, *Nature* **442**, 164–171 (2006)
- 53.136 J.R. Wolpaw, G.M. Friehe, V.A. Zerris, C.L. Ojankangas, M.R. Fellows, J.P. Donoghue: Brain-computer interfaces (BCIs) for communication and control: A mini-review, *Suppl. Clin. Neurophysiol.* **57**, 607–613 (2004)
- 53.137 P.R. Kennedy, R.A. Bakay: Restoration of neural output from a paralyzed patient by a direct brain connection, *Neuroreport* **9**, 1707–1711 (1998)
- 53.138 A.B. Schwartz: Cortical neural prosthetics, *Annu. Rev. Neurosci.* **27**, 487–507 (2004)
- 53.139 B. Pesaran, S. Musallam, R.A. Andersen: Cognitive neural prosthetics, *Curr. Biol.* **16**, R77–R80 (2006)
- 53.140 G.E. Loeb, F.J. Richmond, L.L. Baker: The BION devices: injectable interfaces with peripheral nerves and muscles, *Neurosurg. Focus* **20**, E2 (2006)
- 53.141 R.B. Stein, V. Mushahwar: Reanimating limbs after injury or disease, *Trends Neurosci.* **28**, 518–524 (2005)
- 53.142 P.F. Pasquina, P.R. Bryant, M.E. Huang, T.L. Roberts, V.S. Nelson, K.M. Flood: Advances in amputee care, *Arch. Phys. Med. Rehabil.* **87**, S34–S43 (2006), quiz S44–S45
- 53.143 I. Austen: Computerized control is next for artificial limbs (New York Times, January 3, 2002)
- 53.144 A. Basmajian, E.E. Blanco, H.H. Asada: The marionette bed: Automated rolling and repositioning of bedridden patients, presented at *IEEE International Conference on Robotics and Automation: ICRA'02* (2002)
- 53.145 F. Kasagami, H. Wang, I. Sakuma, M. Araya, T. Dohi: Development of a robot to assist patient transfer, *IEEE Int. Conf. Syst. Man Cybernet.* **5**, 4383–4388 (2004)
- 53.146 G.D. Abowd, M. Ebling, G. Hung, L. Hui, H.W. Gellersen: Context-aware computing, *Pervasive Comput. IEEE* **1**, 22 (2002)
- 53.147 M.J. Covington, W. Long, S. Srinivasan, A.K. Dev, M. Ahamad, G.D. Abowd: *Securing context-aware applications using environment roles* (ACM, New York 2001)
- 53.148 F.H. Wilhelm, W.T. Roth: Ambulatory assessment of clinical anxiety. In: *Ambulatory assessment: Computer-assisted psychological and psychophysiological methods in monitoring and field studies*, (Hogrefe & Huber, Göttingen 1996) pp. 317–345
- 53.149 E. Dishman: Inventing wellness systems for aging in place, *Computer* **37**, 34 (2004)
- 53.150 M. Zinn, B. Roth, O. Khatib, J. Salisbury: A new actuation approach for human friendly robot design, *Int. J. Robot. Res.* **23**, 379–398 (2004)
- 53.151 M. Nokata, K. Ikuta, H. Ishii: Safety evaluation method for rehabilitation robotics. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer-Verlag, Berlin 2004) pp. 187–198
- 53.152 N. Tejima: Risk reduction mechanisms for safe rehabilitation robots. In: *Advances in Rehabilitation Robotics*, ed. by Z. Bien, D. Stefanov (Springer-Verlag, Berlin 2004) pp. 199–207
- 53.153 H.F.M. Van der Loos: Design and engineering ethics considerations for neurotechnologies, *Cambridge Quart. Healthc. Eth.* **16**, 305–309 (2007)
- 53.154 G. Veruggio: The Roboethics Roadmap: <http://www.roboethics.org/site/modules/mydownloads/visit.php?cid=1&lid=37>, (2006)
- 53.155 J.F. Engelberger: *Robotics in Service* (MIT Press, Cambridge 1989)
- 53.156 Z. Bien, D. Stefanov: *Advances in Rehabilitation Robotics* (Springer, Berlin 2004)
- 53.157 E. Prassler, G. Lawitzky, A. Stopp, G. Grunwald: *Advances In Human-Robot Interaction* (Springer, Berlin 2004)
- 53.158 D. Reinkensmeyer, P. Lum, J. Winters: Emerging technologies for improving access to movement therapy following neurologic injury. In: *Emerging and Accessible Telecommunications, Information and Healthcare Technologies: Engineering Challenges in Enabling Universal Access*, ed. by J. Winters, C. Robinson, R. Simpson, G. Vanderheiden (RESNA, Arlington 2002) pp. 136–150
- 53.159 P. Dario, M.C. Carozza, E. Guglielmelli: *Auton. Robot.* **15**, 1 (2003)
- 53.160 J.E. Speich, J. Rosen: Medical Robotics <http://brl.ee.washington.edu/publications/Rep178.pdf>. In: *Encyclopedia of Biomaterials and Biomedical Engineering* (Marcel Dekker, 2004)
- 53.161 D.J. Reinkensmeyer, J.L. Emken, S.C. Cramer: Robotics, motor learning, and neurologic recovery, *Ann. Rev. Biomed. Eng.* **6**, 497–525 (2004)

Domestic Robotics

Erwin Prassler, Kazuhiro Kosuge

Who would not want to have a robot at home that vacuums the house, cleans the kitchen or the bathroom, loads or unloads the dishwasher, or polishes the shoes? In spite of the hundreds of millions of potential customers and users surprisingly few such robots exist. In this chapter, we first look into what it means not only to develop but also to commercialize a domestic robot. Using domestic cleaning robots as a representative example we look into the task details and its context. We also discuss the economic context and the market situation, and the technical challenges which slow down the triumphal procession of domestic robots. We will then have a look at the latest developments of domestic floor cleaning robots, robotic pool cleaners, and window cleaning robots. The survey of domestic cleaning robotics concludes with an outlook to new technologies that might help to solve some of the problems discussed at the beginning. The subsequent section then gives an account on the state of the art in robotic lawn mowing. The Section *Smart appliances* briefly surveys the latest developments in ironing robotics, intelligent refrigerators, and digital wardrobes. The Section *Smart homes* looks into a selection of ongoing and completed research projects in the field of smart environments and smart homes.

54.1	Cleaning Robots	1254
54.1.1	The Task and Its Context.....	1254
54.1.2	Technical Challenges.....	1255
54.1.3	Domestic Floor-Cleaning Robots.....	1258
54.1.4	Pool-Cleaning Robots	1266
54.1.5	Window-Cleaning Robots.....	1267
54.1.6	Old Problems, New Technologies	1269
54.2	Lawn-Mowing Robots	1271
54.3	Smart Appliances	1273
54.3.1	Ironing Robots	1273
54.3.2	Intelligent Refrigerators	1274
54.3.3	Digital Wardrobes.....	1274
54.4	Smart Homes	1275
54.5	Domestic Robotics: It Is the Business Case Which Matters	1279
54.6	Conclusions and Further Reading	1280
	References	1280

The section *Domestic robotics: It is the business case that matters* finally concludes with a contemplation of the market situation for domestic robots, business models, and some crucial insights into the commercialization of service robots.

The dream of having a robot in everybody's home is as old as the word *robot* itself. In Karel Capek's famous play *Rossums Universal Robots* there was already a commercial (poster) advertising for a personal robot: "... Cheap labor. Rossum's Robots. Robots for the tropics. 150 dollars each. Everyone should buy his own robot. Do you want to cheapen your output? Order Rossum's Robots."

Vacuuming the house, cleaning the kitchen and the bathroom, cleaning up the chaos in the children's play-

room, loading and unloading the dishwasher or the laundry machine, polishing the shoes, doing the ironing, stowing away the content of the shopping basket: the list of applications of robots in our homes seems endless. So the question seems appropriate: where are all these smart mechanical helpers that can take care of all these unpleasant tasks? Isn't there a huge market for such devices? Almost everybody would buy one.

There is good news and there is bad news regarding these questions. The good news is: domestic robots are

coming. The bad news is: they are coming very slowly, some of them may be more expensive than many people would like, and most of them will not be the 100% substitute of a robot housemaid or butler, which everybody would like to have.

In this chapter, we will present the state of the art in domestic robotics. We will describe some of the most recent developments in domestic cleaning robotics and a number of other smart appliances, including robotic lawn mowing, ironing robots, and digital wardrobes. We also include a section on smart homes. This may be considered as a borderline area of domestic robotics. However, since smart homes often use sensor, actor, and communication technologies which are very similar to that in *regular* domestic robotics, it is certainly more appropriate to include this topic rather than to exclude it.

Ideally, we would not only present the latest developments in all of the applications and areas above, but also look deeper into the task context, the economic context and market situation, and the fundamental technical problems and challenges. Ideally, we would also identify the emerging key technologies for each of these areas. This may, however, get a bit out of hand and also lead

to many redundancies. For example, the technical problems and challenges for domestic cleaning are not that different from the problems for robotic lawn-mowing. The same holds for the economic situation. So for the sake of a comprehensive treatment we provide a deeper investigation of these problems only for domestic cleaning robotics and confine ourselves to a report on the latest developments for the remaining applications.

Furthermore, since domestic cleaning robotics and robotics for professional cleaning are not entirely different subjects, we will try to provide a broader picture of the technical problems and the economic situation covering both areas. We will see that, especially for professional cleaning robots, the market analysis is somewhat easier to capture, since there are better statistics and more concise business models. Domestic cleaning and domestic robotics in general is still a bit of a gadget market, which is difficult to analyze and predict. So, the excursion to the professional application, while we are contemplating the more economic aspects of domestic and cleaning robotics, may be forgivable.

A discussion on the technology push, the market pull, and the pitfalls in which technology and business developers can easily be trapped concludes this chapter.

54.1 Cleaning Robots

54.1.1 The Task and Its Context

Task Analysis

On an abstract level, the task appears always the same: clean some workspace in the presence of obstacles. The instantiations of cleaning task, however, may differ significantly from environment to environment and from task context to task context. Assume, for example, the task is to clean a swimming pool. Most swimming pools have a rather simple geometric shape – most of them are rectangular – and hence the cleaning task is straightforward to automatize. Using odometry or some low-cost digital compass the pool cleaner should be able to sense and control its orientation and position. Area coverage is a matter of meandering between the walls at the bottom of the pool until the device is turned off or the battery is empty. Obstacle avoidance is mostly unnecessary in such a setting. Not surprisingly, automatically guided pool cleaners are well-established products, which have been on the market for many years. They are not always called robots though.

Now assume a large facility such as a shopping mall or a hospital or an airport with several floors, endless,

narrow and cluttered hallways and with many people moving around. Cleaning such an environment is apparently a different story. The cleaning robot most likely has to face an arbitrarily structured and cluttered three-dimensional (3-D) environment extending over many rooms and possibly over many floors and levels. *Optimal* navigation and operation basically requires 3-D sensing and 3-D modeling. How else should the robot be able to account for 3-D obstacles and navigate in a collision-free manner. Also area coverage becomes significantly more difficult.

The operation and maintenance of large facilities such as airports, shopping malls, or hospitals often involves work flows with a very tight schedule. Often the facilities are large enough to employ a fleet of cleaning robots rather than just one. Fleet management and multirobot coordination are required for optimal execution of the cleaning task. This is even more so when time matters and cleaning has to take place within small time windows. So the use of cleaning robots in such facilities means more than just turning on an off the robots and charging the batteries. It requires a careful integration of the automated service into a sensitive set of work flows.

The above two instances clearly illustrate the variations between cleaning tasks, and at this point we have not even looked into the specific needs and requirements of the surface which needs to be cleaned. Carpet apparently needs to be treated in a different manner from hard floor coverings such as wood or stone. The former needs to be vacuumed or brushed while the latter often requires wet cleaning. Vacuuming requires a significant amount of energy, which makes autonomous vacuuming by battery-powered cleaning robots almost impossible for large areas, scrubbing requires less energy, however often entails heavier devices, since the robots may have to carry nonnegligible volumes of cleaning liquid and dirty water.

In the following we have tried to list some of the typical *dimensions* of a cleaning task:

- containment of work space (enclosed, open)
- *complexity* of work space (cluttered, uncluttered, narrow, wide, static, dynamic)
- scale of work space (small, medium, large)
- dimension (2-D, 3-D)
- surface structure and orientation (even, uneven, horizontal, vertical, slanted)
- cleaning requirements

Economic Context and Market Situation

According to a figure from 1995 [54.1] the professional cleaning services only in Europe total approximately USD 50 billion per year. It can be expected that this figure has further increased in the past 10 years. Of these USD 50 billion about 78%, or USD 39 billion, account for labor, while the remaining USD 11 billion cover equipment, material, and overheads. All in all professional cleaning is a huge market. If one could only automatize a small fraction of these services it would be a billion-dollar or billion-euro business. These figures do not include the domestic market. In Germany alone there are about 40 million households and each needs to be cleaned and in each there is a vacuum cleaner, which is replaced every 6–8 years. If only 15% of these vacuum cleaners would be replaced by a robotic vacuum cleaner there would be a market volume of 1 million units per year.

Engineers and business people realized this potential a long time ago and have been developing cleaning robots for almost 20 years now. Surprisingly we did not see many of those cleaning robots until recently. Apparently, there must have been some problems, which were not only of technical nature. We will discuss these in greater detail at the end of this chapter, after having pre-

sented what is out there already. It should be mentioned, though, that the situation changed significantly, when a little inexpensive device, more of a toy than a cleaning machine, named Roomba came onto the market in 2002.

54.1.2 Technical Challenges

Besides the economical challenge to identify meaningful business cases, the automation of cleaning by means of robots also poses a number of technical challenges. From a scientific point of view these challenges seem to be (almost) solved, but many solutions are not much more than proofs of concept. These theoretical solutions basically work under laboratory conditions but have not been subjected to industrial conditions or faced any extended field tests under real working conditions. Companies that want to develop products often have to reinvent those solutions and adapt them to industrial needs.

The following list gives a short overview of such technical challenges inherent in the design of cleaning robots for domestic as well as professional use. Since the majority of developments in cleaning robotics have been mobile robots for floor cleaning, the list below may be slightly biased towards this application.

- absolute positioning
- area coverage in unknown, dynamic environments
- sensor coverage for robust obstacle avoidance
- error recovery
- safety
- operation interface/human–robot interaction
- multirobot coordination
- power supply

Absolute Positioning. Knowing its current position is essential for the operation of a cleaning robot that needs to cover thousands of square meters of cluttered work space. A cleaning robot must recover its position at every location in its workspace with a reasonable accuracy no matter how far it has traveled already. A robot which loses its position will not be able to execute its task reliably. For the customer this means that the service cannot be delivered regularly and reliably, which is not acceptable. The same holds for other reliability issues (see below), which can easily become liability issues.

There are a number of solutions to the absolute positioning problem. Landmark-based position estimation using passive artificial or natural landmarks is one approach to solve the problem. Position estimation with

active beacons (sonar, IR, radio) is another one. Both approaches are particularly suited for applications where the position has to be estimated sufficiently accurately over arbitrary distances.

Although positioning technology has matured sufficiently enough to enable reliable solutions not only for indoor but also for outdoor applications, one can find significantly more proofs of concepts than off-the-shelf solutions. This holds even more so for solutions with a reasonable price-to-performance ratio. In a later section we will briefly discuss some recent approaches based on smart sensor networks and radiofrequency identification (RFID) technology which may fill this need.

For a detailed discussion and comparison of the above and other approaches we refer to Chap. 37 *Simultaneous Localization and Mapping* or [54.2], where the latter is an excellent, but somewhat outdated state-of-the-art survey on robot positioning.

It is worth mentioning that existing domestic cleaning robots operate almost entirely without absolute position information. Without reliable position information, however, these robots cannot move in a deliberate manner. They can only move randomly in a *bang-and-bounce* mode and/or execute some hard-coded motion patterns. *Bang and bounce* means that the robot moves until it *bangs* into or sees an obstacle and then *bounces off* like a ball by turning around and moving on in an opposite direction. The area-covering performance of such an approach is rather poor (see the next paragraph).

Area Coverage. Motion planning for covering an unknown dynamic environment and absolute position estimation are key functions for systematically cleaning a workspace. The coverage problem – by nature a geometric problem – has been intensively studied in the literature (see [54.3] for a good overview) and quite a few interesting solutions have been proposed even for unknown environments. What is not straightforward at all is the transfer of these solutions from 2-D simulation environments into 3-D real world environments. Particularly the assumptions regarding the perception of the environment and the sensing modalities are often far more ideal than what has to be faced in the real world.

Again it should be noted that available domestic cleaning robots do not provide any systematic coverage of their workspace. They combine random motion with hard-coded motion patterns to achieve some minimal coverage. Given the moderate cost of these devices such a solution is acceptable for many private customers. For professional applications coverage by random mo-

tion has a far lower acceptance, or it may not be accepted by professionals at all. Professional cleaning has a strong association with a certain, guaranteed degree of coverage – it does not necessarily have to be 100% – and not with random motion.

Sensor Coverage. A comprehensive perception of the robot's surrounding is essential for safe, collision-free motion and also for the observation of unknown parts of the environment. So it is important for the safety of the robot and the surrounding objects or nearby animals or humans beings as well as for the successful completion of a mission or a task. Coverage means that the robot does not only perceive some small limited portion of its workspace – e.g., a 2-D range image taken at a certain height above the ground – but has a perception which enables it to account for every known or unknown obstacle or hazard in its surrounding environment. From an academic point of view this is again an almost solved issue. Sensor coverage in 3-D can be achieved by means of stereo vision or 3-D laser range finders. There is an abundance of literature treating this issue.

What remains to be solved is sensor coverage under everyday conditions including changing and adversarial lighting conditions, surfaces with little or no reflection or with little or no texture, and other unfavorable conditions.

The above holds primarily for cleaning robots for professional use. Cleaning robots for private homes have almost no sensors at all. For a device which must not cost more than say 300 USD, a sensor which costs 20 USD is a very expensive component. So adding more and more sensors to make the robot behave more intelligently is not necessarily a good solution as it may significantly increase the price of the robot.

Error Recovery. Every technical system is susceptible to errors. This seems to be a fundamental principle and no design can prevent this or account for every possible error. What is needed are mechanisms which either allow the robot to recover from possible or frequent failures or reach a failsafe position. A very frequent failure situation for a cleaning robot is being trapped in some obstacle structure. The control system must be able to recognize this and provide some escape mechanism or strategy. Other frequent errors are false sensor readings. The robot should be able to identify whether or not its sensors function properly. In the case of malfunction the faulty components should be switched off.

The ability to recover from errors is desirable for every robot, be it a cleaning robot or not. For a commer-

cial device this ability is, however, not only a desirable system property but also has an important economic aspect. Every malfunction which the device cannot recover from will cause a call to the service hotline and a request for repair or maintenance.

Safety. To operate an automatically guided vehicle such as a cleaning robot in a public environment a variety of machinery directives must be obeyed. According to the standard EN954, for example, every automatically guided vehicle needs a front bumper as a personnel protection device in the main travel direction. If the vehicle can reverse its direction it also needs a rear bumper. There are numerous other directives such as ANSI/ASME B56.5-2004 or EN1525:1998 which have to be taken into consideration and obeyed. An interesting insight into this subject is given, for example, in [54.4].

Domestic cleaning robots, given their low weight and low power, hardly create any danger for themselves or the environment. If they get annoying they are often small and light enough to be even kicked away. So the safety requirements for domestic robots are commonly less demanding than they are for professional cleaning robots. Still, many if not all available domestic robots have safety precautions, which satisfy for example EN954. Many have precautions to prevent falling over a cliff or staircases or being picked up and turned upside down or carried away.

Operation Interface/Human-Robot Interaction. The complexity of the operation interface of a device has a strong influence on its acceptance. Given that cleaning devices are typically operated by nontechnical personnel, the operation interface of a cleaning robot which is supposed to replace an existing cleaning device has to account for the needs and expectations of those users or operators. If the use of a cleaning robot would require special education, its use and acceptance would be severely limited. This suggests that the operation interface, for domestic as well as for professional cleaning robots, should be intuitive and straightforward.

Such a conclusion is certainly not false but also not entirely true. The operation interface of any device should allow the user to advance his/her skills in using the device. It should allow but not urge the interested user or operator to also use advanced features, e.g., advanced control or programming of the device.

The design of the operation interface is also influenced by the operation mode in which the robot is used. A fully autonomous robot may only need an on/off switch and an emergency button, while a teleoper-

ated device may have some sophisticated remote control including a sophisticated graphical user interface (GUI).

Multirobot Coordination. Cleaning a large workspace or large facility may easily exceed the capacity of a single robot in terms of onboard power, or cleaning liquid and other consumables, or time constraints imposed by the facility management. For cleaning large facilities the use of a multirobot system is self-suggesting. This raises a few questions, however.

The first one is task planning and coordination for multiple robots. For this central fleet management is required. The degree of automation provided by such a central fleet management may vary considerably. The fleet management might involve a sophisticated task planner, which autonomously decomposes the entire cleaning task and the workspace into subtasks and sub-workspaces, which are then allocated to single robots. In such a setting the fleet management also needs to control the proper execution, i. e., by monitoring the position of the robots, and provide help in the case of errors. In a less automated solution, the fleet management is just a control center for a human operator. Task allocation and monitoring is then done by the human operator.

The fleet management can also be the *bridge* between the robots and the surrounding facility and the automated components therein, for example, the fleet management could open electric doors or call the elevators to allow the robots to move between several floors.

Another problem when using several robots is caused by active sensors such as sonar, infrared, or laser range finders. A robot might interpret a sensor signal which is actually emitted by another robot as the echo of its own sensors. Such false sensor readings severely affect position estimation, map building, and collision avoidance. Therefore the sensor signals of different robots either need to be synchronized or assigned a unique identifier so that the signals can be uniquely assigned to distinct robots.

Power Supply. Covering an area is not only an algorithmic problem. It requires traveling over considerable distances and therefore leads to a considerable power consumption. Since autonomous motion in a cluttered workspace rarely allows a power cord to be pulled behind the robot, power supply is typically provided by batteries and is typically limited. The limitations are due to the weight and the capacities of today's batteries. Domestic cleaning robots as described below claim to have an average operation period of 30–60 min per

charge. Professional cleaning robots often use regular 24 V lead-acid batteries for cars. They achieve longer operation periods per charge accordingly. The price for that, however, is a significant increase of weight, which in turn increases the requirements for safety precautions.

The limited power which is provided today by common battery technology also has an effect on the cleaning technology which can be used for autonomous cleaning. For example, it is nearly meaningless to design a true robotic vacuum cleaner for professional cleaning of larger areas as the energy consumption of vacuum cleaning is prohibitive. The weight of the batteries and the short operation periods per charge make it almost impracticable to use true vacuuming in cleaning robots. Industrial vacuum cleaners are rarely battery driven but typically have power cords.

In the following section we provide an overview of commercial domestic cleaning robots. Not included in this overview are academic proofs of concepts or industrial prototypes. This is partially due to the fact that academic research in this area has almost disappeared since the release of commercial products. Exempted from this are domestic window-cleaning robots. Although the technology is also available this application is still in its infancy and there are no commercial products.

We have divided our overview of domestic cleaning robots into three major categories: *floor-cleaning robots*, *pool cleaners*, and *window cleaners*. We intend to provide a representative but not complete overview of existing systems.

54.1.3 Domestic Floor-Cleaning Robots

Since the year 2000, more than a dozen domestic floor-cleaning robots have been released and a few more have been announced. In the following, we will glance over a total of 13 domestic floor-cleaning robots which were or are commercially available. We have excluded those that have only been announced but do not seem to be available (yet).

Trilobite 2.0, AB Electrolux (Sweden). In 2001, AB Electrolux in Sweden launched the home cleaning robot Trilobite 1.0. This was a milestone in the history of cleaning robotics. After some first developments of domestic cleaning robots in the beginning of the 1990s and even more developments but not very successful market launches of professional cleaning robots, Trilobite was the first home cleaning robot that became commercially available as a *mass* product. Trilobite uses a very sophisticated sonar system for navigation. This sonar



Fig. 54.1 Electrolux: Trilobite 2.0

system allows Trilobite to sense nearby environment and thereby detect and avoid collisions with obstacles. This is a capability which many of the cheaper systems described below do not have. The sonar system also allows Trilobite to follow the contour of obstacles such as walls. After undocking from its charging station, Trilobite explores its workspace by following the delimiting walls until it returns to its starting point. While on its exploration tour Trilobite integrates its sensor information into a map of the workspace. Having information about its workspace allows Trilobite to show a significantly better coverage performance than can be achieved by a pure random motion. Trilobite is not able to determine its position reliably, however, so it is not capable of real systematic coverage of the workspace. Special magnetic strips can be used to lock Trilobite in a room. They act as a wall and can be placed in doorways and other openings. An infrared sensor allows Trilobite furthermore to discover cliffs and stair cases. The price of a Trilobite 2.0 has dropped from around 1900 EUR at the time of the market launch in June 2004 to 1000 EUR in summer 2006.

Robocleaner RC3000, Kärcher GmbH (Germany). Four years after it was announced in 1999, Kärcher's Robocleaner RC3000 was launched in October 2003. RoboCleaner cleans a room by following a random motion pattern. By so doing it gradually covers the entire area which is to be cleaned. Through a sensor which monitors the pollution of the air sucked in it is even able to detect areas which are particularly polluted. When such an area is discovered RoboCleaner increases its suction power accordingly. It has an average cleaning performance of 15 m²/h. Robocleaner RC3000 does not have a sophisticated sonar system like Trilobite 2.0 but uses only tactile sensors to discover collisions with obstacles. It even has tactile *cat-ear-shaped*

Table 54.1 Domestic cleaning robots in Europe

Manufacturer	Electrolux	Karcher	Zucchetti	Friendly Robotics	RoboMop International
System Market release	Trilobite 2.0 10/2004	RC3000 10/2003	Orazio Plus 04/2004	Friendly Vac 05/2004	RoboMop 03/2005
Cleaning technology	Rotating brush, suction pump	Rotating brush, suction pump	Wet mop, dry mop, vacuuming fan	Rotating brush,	Disposable electrostatic pad
Coverage strategy	Wall-following, random motion with obstacle avoidance	Random motion with bang and bounce, spot cleaning with see-saw motion pattern	Random motion with bang and bounce	Parallel and spiral motion patterns, contour following, bang and bounce	No software controlled motion strategy; works purely mechanically, random motion with bang and bounce
Performance	28 m ² /h	15 m ² /h	–	100 m ² /h	60 m ² /h
Sensors	180° ultrasound sensors (1 transmitter, 8 receivers), infrared cliff sensor, magnetic stripe detector, suspended front shield as contact sensor	Suspended front shield as contact sensor, four IR cliff sensor, pollution	360° contact sensor, stair avoidance sensor	Sonar sensors, touch sensor, cliff sensor	No sensing, no programmed works, purely mechanical
Drive system	Differential	Differential	Differential	Differential	Rolling sphere with eccentric weight
Velocity	0.4 m/sec	0.2 m/sec	–	–	0.3 m/sec
Battery	NiMH	NiMH	2×12 V7 A lead batteries	2×127 AH sealed lead acid	Mignon AA
Automatic recharging station	Yes	Yes	Yes	No	No
Run time	60 min	20–60 min	30–60 min	60 min	30–60 min
Size	Ø = 35.0 cm h = 13.0 cm	Ø = 28.0 cm h = 10.5 cm	w = 37.5 cm l = 50.8 cm h = 19.2 cm	Ø = 43.0 cm h = 33.0 cm	Ø = 28.5 cm h = 8.5 cm
Weight	5 kg	2 kg	13 kg	13.5 kg	175 g
Noise	75 dB	54 dB	–	–	–
Price	1300 EUR	1350 EUR	1750 EUR	1599 USD	10 EUR
Status (06/2007)	Available	Meanwhile sold as SIEMENS VSR 8000	Available	Available	Available

sensors at its top which prevent it from getting stuck under a bed or a couch. While it is in many respects similar to the low-cost systems described below Robocleaner RC3000 has a unique feature which seems to be appreciated by its users: robocleaner very reliably returns to

its docking and charging station, once its battery gets low, which is typically after half an hour of operation. While Robocleaner recharges its batteries, its dust bin is emptied by the docking station which itself is a vacuum cleaner. So Robocleaner RC3000 can work unattended



Fig. 54.2 Kärcher: Robocleaner RC3000

for an extend period of time. With a price of about 1500 USD Robocleaner RC3000 is a rather high-priced device. The Robocleaner RC3000 was discontinued by Kärcher and is meanwhile distributed by Bosch-Siemens-Haushaltsgeräte under their own brand name.

Orazio, Zucchetti (Italy). Unlike Trilobite 2.0 and Robocleaner RC300, which are dry cleaners sweeping and vacuuming the floor, Orazio manufactured by Zucchetti, has an additional wet cleaning mode. Altogether it offers five cleaning modes: *continuous vacuuming*: vacuums continuously without use of detergents or cloths, carpets are recognized and get intensive care; *wet cloth*: floor surfaces are processed with a cleansing cloth dampened by detergent solutions, carpets are considered as obstacles and avoided; *dry cloth*: cleanses surfaces without the use of water or vacuuming, carpets are considered as obstacles and avoided; *wet cloth with vacuuming*: surfaces are processed with the cleansing cloth dampened by detergent solutions, when an obstacle is reached, Orazio backs off and vacuums accumulated particles for several seconds, carpets are considered as obstacles; *dry cloth with vacuuming*: cleanses surfaces without the use of water, when an obstacle is reached, Orazio backs off and vacuums accumulated particles, carpets can be considered as obstacles and be avoided or recognized for intensive treatment. Orazio uses only tactile sensing. If it bumps into an obstacle it bounces off and backs up for 5 cm, turns a random angle and then continues. Without any position and range sensing Orazio cannot achieve systematic coverage. Orazio's price ranges from 1299



Fig. 54.3 Zucchetti: Orazio



Fig. 54.4 Friendly Robotics: Friendly Vac

USD for single-floor apartments to 1799 USD for apartments with staircases (staircase detection is included in this case).

Friendly Vac, Friendly Robotics (Israel). Friendly Robotics released a robotic vacuum cleaner called Friendly Vac in 2004. Friendly Vac has a true vacuuming unit and hence claims to have a better cleaning performance than other home cleaning robots. It has an operation time of around 90 min per charge and can clean 100 m² during this time. Friendly Vac has range sensors which enable it to avoid collisions with obstacle and it can also sense stair cases and cliffs. Friendly Vac moves along parallel tracks to cover the cleaning area. If it encounters an obstacle it turns and moves in the opposite direction. With a height of 33 cm Friendly Vac is taller than any other home cleaning robot. It therefore lacks the ability to get underneath most furniture in a home. Also Friendly Vac has no docking station and therefore needs to be recharged manually. Friendly Vac costs 1450 USD.

Roomba and Scooba, iRobot Inc. (USA). The by far most-sold service robot is Roomba developed by iRobot Inc. in Burlington USA. According to an official statement in July 2006, iRobot has sold more than two million units since the product was released in 2002. Both of these events, the market launch of the first version of Roomba in 2002, and record sales figures reached four years afterwards, can certainly be counted as milestones in the history of service robotics. This overwhelming success, however, has little to do with Roomba's performance as a cleaning device – which is not paramount. It exclusively has to do with Roomba's price. Depending on the version, *Roomba Red*, *Roomba Discovery*, *Roomba Discovery SE*, *Roomba Scheduler*, the price for a Roomba ranges between 150 and 330 USD (July

Table 54.2 Domestic cleaning robots in the USA

Manufacturer System Market release	iRobot Roomba 09/2002	Sharper Image eVac 06/2004	Metapo CleanMate QQ-2 03/2006	Black & Decker Zoombot RV501 04/2004
Cleaning technology	Spinning side brush, two counterrotating brushes, suction pump	Rotary brushes, vaccum pump	Vaccum fan, side brush	Vacuum fan, two side brushes
Coverage strategy	Random motion with bang and bounce, contour following, spiral motion	Random motion with bang and bounce, spot cleaning with <i>star</i> pattern	Random motion with bang and bounce, preprogrammed motion patterns	Random motion with bang and bounce, contour following, preprogrammed motion pattern
Performance	–	–	30 m ² /h	–
Sensors	Suspended front shield as contact sensor, IR range sensor, four IR cliff sensors, dust sensor	Touch sensor, cliff sensor	Photosensor for cliff detection	Suspended front shield as contact sensor, cliff sensor
Drive system	Differential	Differential	Differential	Differential
Velocity	0.28 m/sec	–	–	–
Battery	NiMH	NiMH	3 V lithium battery (CR2032)	NiMH
Automatic recharging station	Yes	No	Yes	No
Run time	60–90 min	15–45 min	40–60 min	–
Size	Ø = 35.0 cm h = 8.25 cm	Ø = 31.75 cm h = 14.0 cm	Ø = 35.5 cm h = 9.0 cm	w = 33.8 cm l = 33.8 cm h = 10.2 cm
Weight	2.7 kg	3 kg	2.7 kg	4.5 kg
Noise	80 dB	–	80 dB	–
Price	350 EUR	100 USD	200 USD	100 USD
Status (06/2007)	Available	Available	Available	Available

2006). Regardless of its performance, Roomba was the first robotic appliance to have a price comparable to



Fig. 54.5 iRobot: Roomba

a manually operated device. The robotic technology of Roomba is very simple. Roomba has a differential drive with two rubber wheels and a caster wheel in the front, which give good maneuverability. Roomba’s chassis is suspended. If it is lifted the motors are shut off to prevent damage or injuries. The cleaning mechanism consists of a rotating cylindric brush at the underbody and a spinning side brush on the right. The cleaning mechanism is very sensitive to hairs, carpet tassels, and the like, which can block the mechanism. Roomba has four infrared cliff sensors, which protect it from following down staircases or other cliffs. It has further an infrared sensor which allows it to sense the distance to obstacles on its right and to follow the contour of walls or furniture. Roomba furthermore has a suspended front shield which is used as

a tactile sensor to discover collisions with objects. More recent versions of Roomba have also a dust sensor to recognize areas which are more heavily polluted. Roomba combines several locomotion heuristics to cover a certain area: *following a wall or the contour* of an obstacle using its infrared sensor, a *hard-coded spiral motion*, and a *randomized (zig-zag) motion* which is also used to escape from collisions. Roomba comes with a number of accessories. Roomba's workspace can be confined through so-called virtual walls. These are infrared beams emitted from separate virtual wall units, which can be sensed by Roomba and treated like obstacles. More recent versions of Roomba also come with a charger station, to which the robot returns once its battery gets low. The latest version *Roomba Scheduler* allows the user to program operation intervals in which Roomba shall become active.

A more recent development of iRobot is Scooba, a floor washer. Scooba has a different design and apparently uses a different cleaning technology, but otherwise it uses the same technology. It has the same differential drive system, the same sensor equipment, and uses the same navigation strategies. The price for a Scooba ranges from 300 to 400 USD (July 2006).

eVac Robotic Vacuum, The Sharper Image Inc. (USA).

One of the strongest competitors of Roomba is the eVac Robotics Vacuum by The Sharper Image Inc. released in June 2004. The eVac Robotic Vacuum has an eye-catching design which looks more like the latest design of a modern vacuum cleaner than the flying saucer design of many of its fellows. According to Sharper Image eVac is a true vacuum cleaner with a traditional vacuuming motor. It does not only sweep dirt particles into a dust bin like many other so-called robotic vacuum cleaners. It must be stated though that eVac's vacuuming performance is still far below that of regular AC-powered vacuum cleaners. Except from the 5 inch wheels the drive system is very similar to that of Roomba. Also the navigation system is comparable apart from some details. eVac does not have any range sensors but only relies on tactile sensing. eVac has touch sensors in its front bumper. When this bumper comes into contact with an obstacle such as a wall or a piece of furniture it backs up a little, makes a slight turn, and then continues. Using its bumper eVac can quasi feel its way around an obstacle or along a wall. Like Roomba, eVac combines this contour-following behavior with random motion and with a hard-coded motion pattern. The hard-coded motion pattern consists of parallel tracks along which eVac moves back and forth. Underneath its bumper eVac has two infrared



Fig. 54.6 The Sharper Image: eVac Robotic Vacuum

sensors to detect drops and edges. Unlike Roomba, eVac does not use virtual walls. Instead it comes with four traffic cones which can be used to block a possible escape from the cleaning area. Since the cones can only be sensed by touch they are far less effective than Roomba's virtual walls. The eVac Robotic Vacuum can operate autonomously but can also be controlled by an operator through a radiofrequency (RF) remote control.

While the systems described so far all had some unique features the following two robots seem to be more or less Roomba clones.

CleanMate, Metapo Inc. (USA).

The cleaning robot CleanMate 365, released by Metapo in 2005, has five preprogrammed moving patterns: spiral, special bounce, along wall, s-shape, and polygonal spiral. It iterates through a fixed sequence of these patterns as long as it is in operation, moving at a speed of 35 cm/s. CleanMate 365 has a suspended front shield, which it uses as bumper, and photosensors that detect stairs and prevent CleanMate from falling down. Cleanmate 365's *uniqueness* comes from two minor features. It has a light sensor which enables it to show a helpful behavior. Once its battery gets low CleanMate 365 searches for more light. It moves from darker areas, e.g., from underneath a bed,



Fig. 54.7 Metapo: CleanMate 365



Fig. 54.8 Black & Decker: Zoombot Vacuum

to brighter areas. So, it can be easily found when it has run out of power. Also Cleanmate 365 has an ultraviolet tube attached which can generate ozone to provide disinfection. CleanMate 365 sells for around 200 USD.

Zoombot, Black & Decker Inc. (USA). Nearly unbeatable in terms of price is Black & Decker's Zoombot RV500, which sells for less than 100 USD. In spite of this low price, the robotic technology is basically the same as for a Roomba or for a Karcher Robocleaner RC 3000, which is more than ten times as expensive. Zoombot uses a touch sensor in its front shield for collision detection. After a collision it backs up, turns around, and continues cleaning. It uses motion patterns such as spirals, wall-following, zig-zagging and a random motion to cover the cleaning area. Zoombot has cliff sensors under the bumper to detect drops such as ledges and stairs. It seems, however, that the mechanical design and also the cleaning technology and hence the cleaning performance of Zoombot are clearly inferior to that of all other home cleaning robots discussed so far. It leaves dirt behind and moves considerably slower than its competitors. Zoombot, however, is not the cheapest cleaning robot, as we will see in the next paragraph, although this even cheaper fellow hardly deserves to be called a robot.

RoboMop, RoboMop International (Norway). RoboMop is not what we would consider a typical mobile robot. It merely consists of two parts: a self-propelled *robotic ball*, which pushes an aluminum *cleaning frame*. Attached to the underside of the cleaning frame is a disposable electrostatic pad, which picks up dust as RoboMop moves. RoboMop does not have any motion or coverage strategy or algorithm. It keeps rolling straight as long as nothing blocks its motion. Once it hits an object it physically bounces off the surface like a ball and then keeps moving again. So there is no robotic in-



Fig. 54.9 RoboMop International: RoboMop (RoboMaid in the US)

telligence whatsoever behind RoboMop. RoboMop is really just a self-propelled ball or sphere. There are no sensors which prevent RoboMop from falling down staircases or becoming hung by a cord. However, in spite of the obvious shortcomings of RoboMop it is still an amazing device. It shines in terms of the simplicity of its technology, and it sells for less than 30 EUR. For this amount, the price-to-performance ratio is almost unbeatable, even if it gets stuck every once in a while and cannot vacuum the carpet. The maintenance is manageable and so is the number of components which can break. One simply must not step upon it.

Ottoro, Hanool Robotics Inc. (South Korea). The Cadillac amongst the home cleaning robots is Ottoro made by Hanool Robotics and released in 2003. This comment holds in any respect, including its price of 3400 USD. Ottoro has the most luxurious sensor configuration of all the robots described in this chapter. It has two digital cameras onboard. One of them is specifically devoted to estimate Ottoro's position using a light pattern projected onto the ceiling by Ottoro's base and charge station.



Fig. 54.10 Hanool Robotics: Ottoro

Table 54.3 Domestic cleaning robots in Asia

Manufacturer System Market release	Hanool Robotics Ottoro 03/2002	LG Roboking 08/2005	Samsung/Hauzen VC-R560 07/2006	Yunjin Iclebo 07/2005
Cleaning technology	210 W vacuum pump with revolving suction tool in front	120 W vacuum pump		Main brush, side brush antibacterial filter
Coverage strategy	Systematic coverage by moving along parallel lines and wall following (using precise positioning)	Option: systematic coverage by moving in a lattice pattern, or random motion, or spiral motion pattern	Draws a 3-D map of the environment to identify its relative location, enabling faster and more efficient cleaning of a defined area	Heuristic patterns: random, circular, zig-zag, wall following
Performance	–	–	25 m ² /h	–
Sensors	2 precision cameras (top and front) to sense location and surrounding environment; position estimation using laser projection of light pattern; 12 pairs of ultrasonic; air bumper around chassis	Gyroscope, 7 infrared, 2 wheel sensors, 3 bump sensors, 1 wheel sinking sensor, 3 cliff sensors	–	7 infrared sensors, bump sensors, 3 cliff sensors; safety sensor (shut off when lifting)
Drive system	Synchro drive (3 aligned wheels with synchronized orientation change)	Differential	Differential	Heuristic patterns; random, circular, zig-zag, wall following
Velocity	0.3 m/sec	0.3 m/sec	0.4 m/sec	0.3 m/sec
Battery	Lithium polymer	Lithium polymer	–	Lithium ion
Automatic recharging station	Yes	No	Yes	No
Run time	60 min	70 min	–	150 min
Size	$w = 46.0$ cm $l = 60.0$ cm $h = 28.0$ cm	$\varnothing = 34.0$ cm $h = 13.5$ cm	$\varnothing = 36.0$ cm $h = 13.0$ cm	$\varnothing = 35.0$ cm $h = 9.0$ cm
Weight	15 kg	–	6 kg	4.1 kg
Noise	60 dB	–	–	59 dB
Price	3400 USD	900 USD	800 EUR	530 USD
Status (06/2007)	Available (only in Asia)	Available (only in Asia)	Available (only in Asia)	Available

With this, Ottoro is possibly the only domestic cleaning robot that can estimate its position with respect to the base’s reference system. According to the product specifications Ottoro senses its position with an accuracy of ± 3 cm. A problem of course arises if Ottoro leaves the area from which the light pattern on the ceiling is visible. Besides the two cameras Ottoro uses 12 pairs of ultrasonic sensors for safe navigation and obstacle avoid-

ance. Furthermore *highly sensitive* air bumper sensors are wrapped around Ottoro’s chassis. With the ability to estimate its position with a reasonable accuracy, Ottoro does not have to rely on random motion or other motion heuristics to cover the cleaning area. Based on a map of the work space Ottoro can plan a systematic cleaning path, monitor its execution by its location sensor, and also register the cleaned area to avoid multiple vis-

its and inefficient coverage. Ottoro not only has a unique sensor configuration but also uses a unique drive system and a suction spout (*Ottoro's hand*). Its drive systems consists of three *moveable* wheels (three-wheel synchro-drive), which allows Ottoro to move instantly in any direction from any location. Its suction spout allows Ottoro to clean corners and spots which are out of reach for any other domestic cleaning robot.

In summary, Ottoro is a rather unique exemplar amongst existing domestic cleaning robots as it makes full use of the state-of-the-art technology in mobile robot navigation. Apparently this does not come for free. It is unknown how many units of Ottoro have been sold.

RoboKing V-R4000, LG (South Korea). The second out of the four South-Korean-made domestic cleaning robots is Roboking (model name: V-R 4000). Roboking's sensory equipment and design is not as luxurious as that of Ottoro and not as expensive, but is still impressive. Roboking is controlled by a 32 bit digital signal processor (DSP). It uses a gyro sensor to monitor its motion and control its position. This is a valuable extension to position control based purely on odometry, but it does not give absolute position like Ottoro's localization system. Besides the gyroscope Roboking has seven infrared sensors for obstacle avoidance, three cliff sensors, two shaft encoders, three bumper sensors, a wheel-sinking sensor, a dust amount and dust bag sensor, and two rotational brush driving sensors. Depending on the *cleaning circumstances* Roboking can switch between four cleaning patterns: *immaculate cleaning/gyro matrix moving*, in which the robot moves in a left-to-right lattice pattern followed by an up-and-down lattice pattern; *perfect cleaning/variable matrix moving*, in which the robot moves in a small lattice pattern and expands its motion pattern when reaching larger free space; *intensive cleaning/spiral moving*, in which the robot spirals around a specific space to clean it; *fast cleaning/random moving*, in which the robot moves randomly and not on a fixed pattern. LG further advertises the powerful



Fig. 54.11 LG: RoboKing V-R4000



Fig. 54.12 Samsung: VC-RP30W Robotic Vacuum

brushless direct current (BLDC) suction motor (120 W) and the lithium polymer (Li-Po) batteries of Roboking. Roboking sells for around 900 USD.

VC-RS60 Robotic Vacuum, Samsung (South Korea). Not much information is available on the VC-RS60 Robotic Vacuum announced by Samsung in November 2003. It has a webcam which can even be accessed through the Internet, so that the user can control the operation while elsewhere. The VC-RS60 can be programmed to work at specific times like Roomba Scheduler. It is said to create a 3-D map of the environment to identify its relative location, enabling faster and more efficient cleaning of a defined area. In July 2006, Samsung released a revised model, the VC-RS60, which even offers voice control. The new model, which seems to be available only for the Korean market at this time, sells for approximately 750 USD.

Iclebo, Yujin Robotics (South Korea). While its three compatriots were distinct enough in their technical design not to be put into one class with the Roomba family, the design of Iclebo and its upgrades is very similar to that of Roomba. It combines a number of heuristics such as bump and bounce, wall following, moving on parallel tracks or on a circular trajectory. It has a set of seven infrared sensors to monitor the surrounding workspace and uses three optical cliff sensors to keep it away from



Fig. 54.13 Yujin Robotics: Iclebo

staircases and ledges. Iclebo has three filter systems: an antibacterial filter, an electrostatic filter, and a semi-high efficiency-particulate airfilter (HEPA). Iclebo was released by Yujin Robotics in summer 2005 and sells for around 500 USD.

54.1.4 Pool-Cleaning Robots

Domestic cleaning robots, in spite of promising sales figure, are still struggling to get rid of the reputation of being exotic devices or toys and to become regular appliances. Pool-cleaning robots never really had such problems. Robotic pool cleaners have been established products for years. This may be due to the fact that the challenge of cleaning a rectangular pool is rather modest and so is the *robotic technology* used in robotic pool cleaners. Given the liquid surrounding there is not much sensing other than motor encoders and measuring the motor current for obstacle detection. In the following we only describe the three cleaners which are the most established.

Aquabot, Aqua Products (USA). Aquabot is an automatic pool cleaner for residential pools. It uses two sealed high-quality motors. One motor causes the drive belts and drive tracks to move the unit. They also cause the front and rear scrubbing brushes to clean the pool surface, walls, and steps. The second motor is the pump motor, which generates extremely powerful suction so that Aquabot is not only able to filter the water but can also climb up the pool walls to the waterline. Aquabot has an external power supply through a floating power cord. Aquabot and Aquabot Turbo alternate between two cleaning patterns. One is a zigzag motion, where the wall is used as a navigational aid. Whenever Aquabot hits the wall of the pool it reverses its direction and moves to-

wards the pool center at a certain angle. On each pass it covers roughly 60% of the pool area. The second pattern is a rectangular meander pattern, where again the wall is used as a navigational aid. Aquabot has a sensor to detect obstacles such as walls or heavier objects at the bottom of the pool appearing in front of it. Once it discovers a collision it reverses direction and continues moving in the opposite direction. Aquabot (Turbo) has a cleaning performance of 315 square meters per hour and sells for approximately 850 USD.

TigerShark Pool Cleaner, Aquavac (USA). TigerShark is the basic model of a product line of *xShark* robotic cleaners by Aquavac Systems. The basic design of TigerShark is very similar to that of Aquabot. It is equipped with two polyurethane drive tracks. TigerSharks motors – one for locomotion, one for the suction unit – are sealed in a motor unit. It has a *calculated* ground speed of 15 meters per minute and a work cycle of 5 h. TigerShark’s cleaning unit consists of a suction pump with a suction rate of 300 liters per min and a removable, reusable, cartridge filter. TigerShark’s suction power creates enough adhesion to let it climb up the walls of the pool. Very little information is available on TigerShark’s control system and its coverage strategy. According to the user manual TigerShark is equipped with an adaptive seek control logic (ASCL) microprocessor which can sense the pool contour and calculate an efficient motion pattern to achieve coverage. Unfortunately, it remains a secret exactly how the contour is sensed and how the motion pattern is calculated. A reasonable guess is that both Aquabot and TigerShark measure the motor current to sense an obstacle in their way. Like Aquabot, TigerShark sells for around 850 USD.



Fig. 54.14 Aqua Products: Aquabot



Fig. 54.15 Aquavac: TigerShark Pool Cleaner



Fig. 54.16 Maytronics: Dolphin Diagnostic 2001

Dolphin Diagnostic 2001, Maytronics (Israel). Dolphin Diagnostic 2001 is the best-selling member of a whole family of Dolphin automatic pool cleaners manufactured by Maytronics in Israel. Dolphin Diagnostic 2001 seems to differ from TigerShark by Aquavac only in color. A careful look into the user manuals and into the technical drawings is required to identify differences in the mechanical design. Dolphin Diagnostic 2001 is slightly lighter than TigerShark and has slightly less suction power. It has a ground speed of 15 m/min and a coverage rate of 350 m²/h. Dolphin Diagnostic 2001 is capable of exploring the size and shape of a pool and calculating an efficient cleaning pattern. As for the TigerShark it remains unclear how the exploration and calculation of a cleaning pattern work. The manufacturer, Maytronics, does not reveal any information on this. Dolphin Diagnostic 2001 has an integrated air sensor which will detect if the cleaner has left the water, turn itself around, and crawl back into the water to continue cleaning. Maytronics advertises Dolphin's diagnostic capabilities. The cleaner runs a constant self-diagnostic program and allows technicians to instantly download information about the operation of the unit for in-the-field diagnostics and on-site repair. Dolphin automatic pool cleaners range from 800 USD for residential cleaners to 5000 USD for professional cleaners used for public pools.

54.1.5 Window-Cleaning Robots

While robotic pool cleaners and also domestic floor-cleaning robots have become established and accepted products, the automation of another area of domestic cleaning is still in its infancy: window cleaning. The reasons for this are not too difficult to find. Windows in private homes are cleaned far less often than the floor. Who would buy a robot for automatizing a task

which needs to be done every once in a while and how much would he or she pay for it? The answer is certainly frustrating for somebody who wants to develop window-cleaning robots for domestic applications. While floor cleaners do not bother about gravity and falling down unless they are near staircases or ledges, gravity is an essential problem for window-cleaning robots, and the solutions are usually not very cheap. Special mechanism have to be designed for secure motion. Typically special tether mechanism prevent the robots from falling. Special locomotion mechanisms have to create enough adhesion force to hold a robot attached to a flat, vertical, damageable surface such as glass and at the same time move the body up and down and sideward. These mechanisms have to be small and light and create enough adhesion forces and must have low energy and resource consumption. The cost for those mechanisms are typically far beyond the price even of expensive appliances. This may explain why there are no commercial domestic window-cleaning robots available at present.

Robots which have to climb up even vertical surfaces often use caterpillar drives which are equipped with passive or active suction cups. Passive means that the system does not actively create a vacuum in the cup. Rather a small valve aerates or seals the suction cup depending on the position of the cup along the drive. Drives with passive suction cups have the advantage of moderate energy consumption. They have, however, one severe disadvantage. They tend to lose their adhesion after a while. The reason for this is a torque which acts on the center of gravity of the system. Due to this torque there is a traction force acting on the upper cups while at the same time pressure is exerted on the lower cups. Without any attractive force acting on the upper cups the adhesion there gets weaker and eventually the system falls. Therefore passive suction cups are rarely practicable.

An apparent solution to this problem is the use of active suction pumps, which generate a vacuum under the upper suction cups. This solution prevents the system from falling. However, supplying the vacuum to the cups makes the system significantly more complex, heavier, and larger. Researchers at Fraunhofer IPA [54.5] have therefore invented a smart solution which gets by with passive cups, but gets around the problem of decreasing adhesion. The solution uses a spacer at the rear of the vehicle. This spacer (see Fig. 54.17) neutralizes the torque around the center of gravity which is typical for a systems with passive suction cups. The spacer causes a traction force which acts on the lower suction cups. This traction force creates a torque around the spacer,

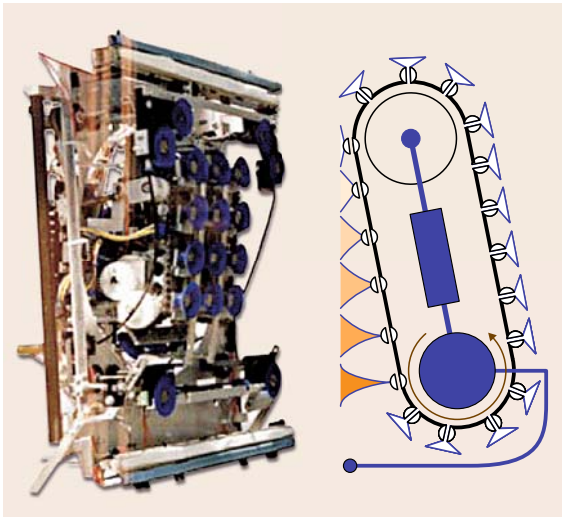


Fig. 54.17 Fraunhofer IPA: climbing robots with passive suction cups

which counteracts the torque around the center of gravity and also causes a pressure on the upper suction cups.

RACoon, Fraunhofer IPA/Procter & Gamble, (Germany). RACoon is a window-cleaning robot which was developed in a joint venture between Procter & Gamble and Fraunhofer IPA and was presented during the Hannover Fair 2002. RACoon uses passive suction cups with a spacer as described above. RACoon was developed as a demonstrator for automatic domestic window cleaning. Given its size and weight, the design shown in Fig. 54.18 below was not really intended to become a product.



Fig. 54.18 Fraunhofer IPA: the window cleaner Raccoon



Fig. 54.19 Fraunhofer IPA: bottom view of QUIRL



Fig. 54.20 Fraunhofer IPA: QUIRL design study

QUIRL, Fraunhofer IPA, (Germany). With the window cleaner QUIRL Fraunhofer IPA presented a successor of RACoon with a totally revised design. In QUIRL, the number of components, the weight, and the size of the system were significantly optimized, as can be seen in the figures 54.19 and 54.20. The main functions *cleaning*, *holding*, and *moving* were unified in one single component. The prototype of QUIRL consists of two vacuum cups which are attached to a common frame and which are driven by two separate motors and rotate independently of each other. The overall motion of QUIRL can be controlled by selecting the velocities and rotational directions of the vacuum cups. If the motor of one vacuum cup is turned off and the cup does not rotate, QUIRL rotates around this fixed cup. If both motors and cups rotate in the same direction this leads to an overall rotation of QUIRL about its vertical axis. If both drives rotate in the opposite direction at exactly the same velocity then QUIRL makes a linear motion. If both drives rotate in the opposite direction but their velocities are not identical then the translational motion is superimposed by a rotational motion and QUIRL moves along a curved trajectory. In order to clean the surface some cleaning mechanism or tool needs to be fixed to the vacuum cups. By attaching, for example, specific cleaning

towels in the cups the abrasion effect is increased and a very good cleaning performance can be achieved. In summary, by implementing the cleaning, holding, and moving function by one single mechanism the system complexity of QUIRL, its weight, its size, and not to forget its cost could be significantly reduced. QUIRL is described in more detail in [54.5]. There is only one problem: QUIRL is still waiting to become a product.

54.1.6 Old Problems, New Technologies

In this section we discuss some new technologies which seem to have the potential to solve some of the pressing technical issues discussed in Sect. 54.1.2. The following is a list of the most promising of such enabling technologies:

- low-cost (3-D) sensing (photonic mixer device (PMD) or other technology in mass manufacturing)
- low-cost absolute localization (local positioning systems), sensor networks
- new energy concepts such as fuel cells

Should the promises come true and the potential of these enabling technologies materialize then they might in fact lead to a considerable increase of the acceptance and use of cleaning robots in domestic as much as in professional environments.

3-D Sensing (at Low Cost). We have learned above that sensor coverage and a comprehensive perception of the workspace are fundamental for the safe operation of a cleaning robot. Sensor coverage at present is certainly possible but at a high price. This situation seems to be changing with the arrival of small 3-D time-of-flight range cameras such as the SwissRanger SR-3000 [54.6]. These cameras use highly sensitive, custom-made complementary metal-oxide-semiconductor (CMOS)/charge-coupled device (CCD) image sensors and measure the time of flight (TOF) of an emitted light wave in the near-infrared spectrum.

The development of this sensor technology was primarily driven by the automotive industry for safety applications. Hence the expectation is that there is a mass market and the sensor can be produced and distributed at a reasonable price. It was predicted that the price of these 3-D range cameras would come down to the level of an expensive webcam. At present the price for a single unit is still in the order of 5000 EUR, which means that its enabling features are somewhat hampered by the cost.

Absolute Localization (at Low Cost). A radical solution to the absolute positioning problem is to cast the workspace with a sufficiently large and dense network of artificial markers or beacons, which can be sensed by the robot nearly from everywhere in the workspace. An overview of such approaches and systems developed in the recent past is given in [54.7].

A very recent approach has been developed in a joint venture of two German companies: Vorwerk Teppichwerke, a carpet manufacturer, and InMach Intelligente Maschinen, a developer of robot control and navigation systems. This approach uses a so-called *smart floor* [54.8] to support robot position estimation and navigation over large distances. The core idea of the smart floor is to distribute and integrate RFID transponders in the floor in an area-covering manner and use the transponders for various purposes. Since integration into the floor covering itself would complicate the manufacturing and increase the price of the floor covering and would have to be implemented separately and specifically for every type of floor covering, Vorwerk Teppichwerke has developed a so-called *smart underlay* (see Fig. 54.21).

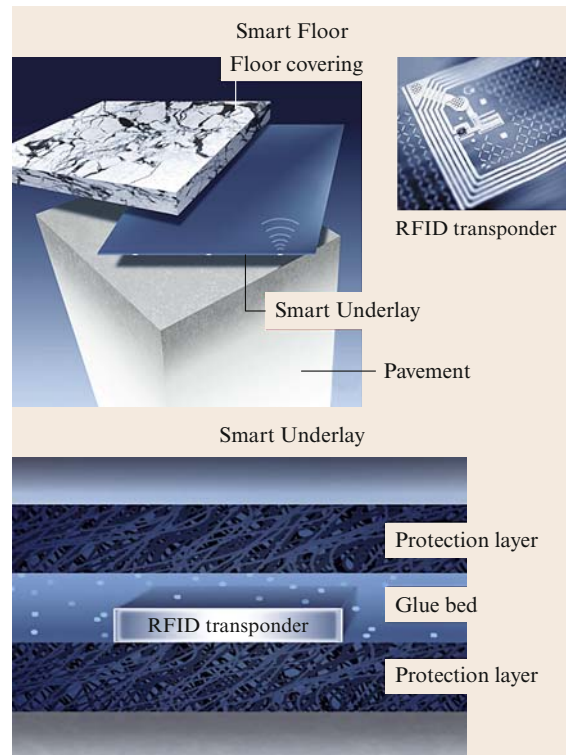


Fig. 54.21 The design of smart underlay and smart floor



Fig. 54.22 CeBIT 2006: smart underlay placed under carpet, laminate, tile, and PVC



Fig. 54.23 Professional cleaning robot Robo40 navigating on smart floor

The smart underlay is a separate mat-like textile, in which the **RFID** transponders are integrated, typically in a regular grid. This underlay can be placed nearly below every nonmetallic floor covering. During CeBIT 2006, the world's largest information technology (IT) fair, the smart underlay concept was demonstrated to the public. The smart underlay was placed under four different floor coverings: carpet, laminate, tile, polyvinyl chloride (PVC) (see Fig. 54.22). The Swiss–German cleaning robot Robo40, shown in Fig. 54.23, is the first one to operate and navigate on such a smart floor. It is equipped with **RFID**-based navigation system which uses the **RFID** tags integrated into the underlay to estimate the position and control the locomotion of the robot [54.9].

The above system is not per se low-cost or much cheaper than other ones involving expensive sensors and sensing. That is true even if the cost of a **RFID**

tag, which is around 30 to 40 EUR cents at present, are dropping further. After all, several thousands of square meters may have to be equipped with **RFID** tags or other sensors or markers with additional costs of up to several euros/dollars per square meter. Whether or not such an investment pays off and whether the price-to-performance ratio is acceptable for the customer very much depends on the specific application. In many applications only the automation of cleaning and some logistic and transportation functions justify and pay-off the investment in a smart infrastructure or a smart floor. Often there is a range of additional services or functions facilitated by such a smart environment, e.g., location-based services and other ubiquitous computing functions, which yield an even better return on investment. A common feature of solutions such as those described is undoubtedly that they enable robust and reliable absolute position estimation over arbitrary distances.

An alternative solution to wide-area indoor localization is the NorthStar system developed by Evolution Robotics. The idea underlying NorthStar is to project infrared light patterns onto the ceiling above the robot's workspace and use these patterns for position estimation very much like old sailors used the stars in the sky, including the north star, to determine their positions in the vastness of the seas. NorthStar consists of two types of components: one or more IR projectors, which emit a collimated IR beam with a unique signature which leaves a unique IR light pattern on the ceiling, and a detector which uses a triangulation algorithm to estimate the position and heading in relation to the IR light pattern on the ceiling. The basic setup of a NorthStar system is shown in Fig. 54.24. The distance between

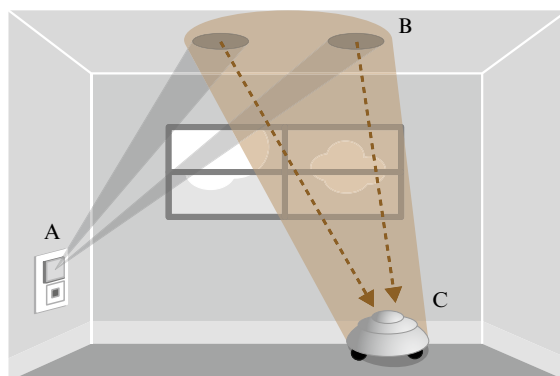


Fig. 54.24 Basic setup of a NorthStar system with an IR projector(s) mounted at fixed positions and a mobile detector carried, for example, by a robot

the detector plane and light plane must not exceed 6 m according to the product specification. Covering a larger workspace requires a rather dense network of projectors to be installed. Given that one projector sells for around 600 USD, this may become quite costly. It may be the case, however, that for a larger installation the price for one projector unit comes down significantly.

New Energy Concepts. In the preceding section it has become evident that energy consumption and power supply have a significant impact on the design and use of cleaning robots. A cleaning device which needs to be

recharged every hour is useless for professional cleaning. Therefore better and more efficient power supply concepts are a burning issue in cleaning robotics.

A promising solution for the power supply problem, although not exhaustively investigated, may be fuel cells. A first practical application has been reported in [54.10]. The Tokyo-based company Sohgo Security has been developing a sentinel robot equipped with a fuel-cell battery that is supposed to work round the clock for one week without a recharge. The fuel-cell battery is to be supplied by Yuasa Corp., a Japanese battery manufacturer.

54.2 Lawn-Mowing Robots

As popular as domestic cleaning robots are robotic lawn-mowers. It may be somewhat surprising that the first commercial robotic lawn mower was released as early as 1995, long before any commercial domestic cleaning robot was in sight. This may be due to the fact that lawn-mowing is a domain where the performance requirements are not as critical and the customer expectations are not as high as in domestic cleaning.

Cleaning robots and lawn-mowing robot share a great deal of technical problems (see Sect. 54.1.2) and also solutions. Apart from the different services, cleaning versus mowing, and the different application-specific processing units, there are just minor differences in the basic system designs. Most lawn-mowing robots have differential drive systems with castor wheels. They use sensor equipment for navigation similar to that of cleaning robots and they use similar coverage strategies, mostly a bang-and-bounce strategy.

Since robotic lawn-mowers work outdoors, they can run away. To prevent this they are often kept enclosed by virtual fences. These are wires buried in the ground that emit an electromagnetic field which can be sensed by the robots and push them back. All robotic lawn-mowers have lift protection as an essential safety mechanism. The cutting mechanism is immediately shut off once the robot is lifted or falls upside down.

Given that the majority of humankind lives in cities, the market size for robotic lawn-mowers is certainly smaller than that for robotic cleaning devices. This may explain why the number of brands of robotic lawn-mowers that are available is also smaller than that of cleaning robots. In the following we describe the three most established commercial system and a fourth system which was introduced recently.

AutoMower, Husqvarna/Electrolux (Sweden). The father of all domestic robots, if not of all commercial service robots, is the predecessor of the AutoMower manufactured by Husqvarna, a subsidiary of the Electrolux group, Sweden. This predecessor, SolarMower, was released as early as 1995: earlier as any other known commercial domestic service robot. Unlike SolarMower, which was powered by a solar panel placed on top of the robot, AutoMower is powered by nickel-metal hydride batteries. When running out of power AutoMower returns to a charging station and recharges its batteries. A normal recharge takes approximately 1.5 h. Fully charged AutoMower can operate for up to 2 h. AutoMower is a lightweight amongst the robotic lawn mowers, weighing only 8.5 kg.

AutoMower's cutting mechanism consists of a rotating disc with three razor-like blades which automatically retract into their mountings if AutoMower is stopped unscheduled, for example, if it hit an obstacle or is lifted. It is important for the proper functioning of AutoMower that the lawn which is to be cut is not too high. This in turn requires a rather regular if not continuous operation of AutoMower. With regular use of AutoMower the



Fig. 54.25 Husqvarna/Electrolux: AutoMower

grass cuttings are short enough to quickly decompose into nutritious compost, so there is no need to remove the cuttings after the lawn is mowed.

AutoMower has no position or range sensors which would allow keeping track of its motion. It uses a random motion pattern to cover its work space and keep the lawn at an equal height. To prevent AutoMower from running away from its workspace a low-voltage induction cable is buried around the lawn. Once AutoMower senses this cable it stops, reverses its direction away from the cable, and then moves on towards the inner area of the workspace. Using this random motion and a bang-and-bounce strategy AutoMower can maintain an area of up to 1500 m². AutoMower is sold for approximately 2000 EUR.

RoboMower, FriendlyRobotics (Israel). Friendly Robotics (formerly Friendly Machines) also started developing a robotic lawnmower *Lawnkeeper* in 1995. Several product iterations finally led to the current product family RoboMower RL 850 and RoboMower RL 1000. With a weight of 22.5 kg without batteries RoboMower is significantly heavier than Automower. It is powered by two maintenance-free 2 × 17 AH sealed lead-acid batteries with a charging time of approximately 20 h.

RoboMower is equipped with touch sensors around its hull. If it collides with an obstacle it senses the contact, stops, turns around, and moves on in the opposite direction. RoboMower has furthermore sensors to detect the wire which is buried around RoboMower's work space. Although it has no position or range sensor RoboMower does not move entirely randomly. By using the buried wire as a reference and moving in a zigzag pattern in the confined area it can achieve a better coverage performance than a pure random motion would allow. To achieve good cutting performance RoboMower still needs to cover its workspace several



Fig. 54.26 Friendly Robotics: RoboMower 850



Fig. 54.27 Zucchetti: LawnBott Evolution

times. Using this strategy RoboMower RL 850 can mow and maintain an area of approx. 1000 m². RoboMower RL 850 without a docking station sells for USD 1000. RoboMower RL 1000 with a docking station sells for 1500 USD.

LawnBott, Zucchetti (Italy). A whole series of lawn-mowing robots have been developed by Zucchetti, Italy. LawnBotts (Professional, Deluxe, Evolution, Quattro) come with slightly different designs, features, and prices but do not substantially differ in their robotics technology. Also, the differences with respect to the previously described systems are minor.

LawnBott's workspace needs to be delineated by either an induction wire buried around the cutting area or by a fence which is at least 10 cm high. LawnBott uses a bang-and-bounce strategy to deal with obstacles. Coverage is achieved by random motion. LawnBott senses the height of the grass. If it discovers a spot with high grass, it stops moving randomly and enters a specific motion pattern. Starting at the spot with the higher grass it moves along a spiral until it reaches an area where the grass has nominal height again.

All LawnBott models are equipped with a wet grass sensor, which discovers when rain starts falling and sends the robot back to its base station. LawnBott's base and recharging station looks like a small garage and accommodates the robot entirely, so it is protected against rain and storms. All LawnBott models have a coverage performance of around 270 m²/h and can cover an area of 3300 m² in total. The price of the LawnBott models ranges between USD 1850 and 2500.

RobotCut, Brill (Germany). A rather recent development with an eye-catching design is RobotCut, developed by the German companies Brill and InMach Intelligente Maschinen and commercialized by Brill. RobotCut was



Fig. 54.28 Brill: RobotCut

presented at GAFA 2006, an international fair for garden equipment, in Cologne, Germany. Unlike its competi-

tors, which all use a rotating disc as cutting mechanisms, RobotCut uses a spindel cutting system, which claims to create a rather uniform cutting image. RobotCut has two rather large propelled front wheels which give the vehicle additional stability. RobotCut has a cutting width of 38 cm. The cutting height can easily be adjusted by screws at the rear castor wheels. RobotCut's brain is the so-called *optimized guidance system*, which enables the robot to move along parallel tracks and avoid random motions. RobotCut is designed to cover an area of 2500 m². RobotCut will be delivered in several versions, with a high-end version also being equipped with sonar for obstacle avoidance. RobotCut is announced for 1500 EUR.

54.3 Smart Appliances

The term *smart appliance*, although well established and widely used, seems to be everything but well defined. A Google search returned no fewer than 27 400 entries for *smart appliance* at the time when this text was written. Very often the term *smart* is used as a synonym for *networked*, referring to devices which are connected to and can communicate with the Internet. Smart appliances are often referred to in the context of home automation and ambient intelligence.

Our understanding of this term is somewhat narrower. What we have in mind when we talk about *smart appliances* are appliances which in one way or another use robotic technology (sensors, actors, smart control systems). Although it might seem to be more appropriate to then talk about *robotic appliances* we refrain from doing so, since we do not see a need to draw a clear line between *robotic appliances* and *smart appliance* at large.

One reason for this is that there are not that many *robotic appliances* other than the cleaning and lawn-mowing robots, which we have already described. In spite of the endless list of tasks which people might want to delegate to some kind of robot we found only two more applications which might be counted as *some kind of robot* in a colloquial sense. In both applications there is a degree of actuation and automation involved, which is enough reason for us to include them in a chapter on domestic robotics. The three applications described in the following three sections are: *ironing robots*, *intelligent refrigerators*, and so-called *digital* or *smart wardrobes*.

54.3.1 Ironing Robots

The plural in the title may raise wrong expectation. During our search we actually came only across a single device which was called as ironing robot even by its inventors: *Siemens Dressman* (also distributed as *Bosch ShirtMaster*).

Dressman, Siemens (Germany). The Siemens Dressman ironing robot complies with the definition of a robot



Fig. 54.29 Siemens: Ironing robot Dressman

only in the wider sense. It is a machine which automates a certain service, which is disliked by many people: ironing. More specifically, Dressman is designed to *iron* shirts. It does so by means of an inflatable torso made of parachute silk material. Once the shirt is stripped over the torso it is inflated with hot air in its interior. This hot air presses and dries the shirt. Creases are removed during this process. The hot air is created in a heater box in the underbody of the device. The torso adjusts perfectly to that of the garment, allowing the pressing of all kinds of shirts, including short-sleeved ones. When the ironing process has finished Dressman exhausts cold air for one minute in order to stabilize the cloth and prolong the ironing effect. The inflatable torso can accommodate shirts in European sizes of 35 to 50 (XS to XXXXL) and in US sizes of 38 to 52 (S to XXL). When fully set up Dressman measures 173 × 36.5 × 45 cm and weighs 28 kg. During operation it consumes approximately 3300 W. Dressman was released in 2004 and sells for around 1000 EUR.

54.3.2 Intelligent Refrigerators

While the plural *ironing robots* in the headline of the preceding section was an overstatement, talking about intelligent refrigerators definitely is not. There are at least four so-called intelligent refrigerators out in the market: LG's *Internet Refrigerator*, Siemens' *CoolMedia fridge freezer*, Samsung's *HomePAD's Refrigerator*, and Electrolux' *ScreenFridge*. As indicated in the introduction to this section, *intelligent* in the context of refrigerators means *networked*. The share of robotic technology involved in an intelligent fridge is negligible. For this reason we only briefly discuss one representative of this class of smart appliances, namely the Electrolux ScreenFridge (see Fig. 54.30), which is one of the latest developments. ScreenFridge has a wireless connection to the Internet and to the TV. Its user interface is a 15-inch touch screen and pop-up keyboard. In addition to Internet, email, phone, radio, and MP3 player ScreenFridge also offers an advanced calendar and video messaging system. In spite of some visionary's expectations, there is currently no intelligent refrigerator on the market which can call the supermarket and order a six pack of beer when it notices that the beer stock gets low. There is no fundamental technical problem to implementing such a vision. What may be a barrier is the lack of an infrastructure to support such a service.

Although we do not want to further elaborate on this, it should be mentioned that large appliance manufactures



Fig. 54.30 Electrolux: intelligent refrigerator *ScreenFridge*

such as LG and Samsung have apparently discovered the Internet for refining their appliances. Both companies do not only offer networked refrigerators but also other devices. LG and Samsung, for example, both offer an *Internet microwave oven*. LG even has an *Internet turbo drum washer* on the market. How big the market for such *intelligent* appliances currently is or in the future will be, is hard to predict. It may not be in everybody's budget to pay around USD 9000, for example, for a ScreenFridge.

54.3.3 Digital Wardrobes

The idea of fully automatizing not only the handling and maintenance of clothing but also to have support in the selection of the right dress for the right circumstances has led to the concept of *digital wardrobes*. Although this is a fascinating idea, which might especially be interesting for one or another bachelor who is in dire need of support in this respect or for those which have a significant collection of clothes and shoes, the implementation of the idea into products is still pending. We have come across only one design study for a digital wardrobe. This design study includes a business plan but otherwise awaits implementation. The design study uses little conventional robotics technology. The emphasis is more on the information technology side rather than on the physical handling.

EDWard – The Intelligent Wardrobe, Technical Univ. Darmstadt (Germany). The appearance of EDWard is

not particularly unique. It does not differ from any regular wardrobe. What makes EDWard unique is what the inventors call *clothes awareness*. EDWard knows about every piece of clothing which is stored in it. This includes shoes, as well as hats and ties. This clothes awareness is implemented by means of a database and a tracking system, which notices whenever an item is taken out of or put into EDWard. The database contains an entry for every piece of clothing stored in EDWard. It also contains information on how pieces of clothing can be combined to form an attractive outfit or for which temperatures or weather conditions they are suitable for. To keep track of what is going in and out of EDWard, every item needs to be marked by an **RFID** tag, which carries a unique identifier and other information which might be better stored directly on the item rather than in EDWard's database. The computer which hosts the database is also connected to an **RFID** reader. Whenever a piece of clothing is taken out or put into EDWard it passes the reader and the antenna attached to it. In this way EDWard maintains a full record of what is stored in it.

Apart from the basic storage and retrieval function the inventors propose a range of additional functions such as search for clothing in the wardrobe, a *tie assistant*, a *shopping assistant*, which cares for compatibility and complementary in the dress and show collection, a *travel assistant*, which proposes the right dressing for the journey depending on the climate and weather con-



Fig. 54.31 Technical University Darmstadt: the intelligent wardrobe EDWard

ditions at the destination and the business to be done, a *washing assistant*, and many more.

As already mentioned there is not too much robotic technology involved in EDWard at present. However, there are quite a few robotic function which could be added, for example a *robotic packing assistant*, which not only proposes clothing for the trip but also packs it into the suitcase.

54.4 Smart Homes

Several attempts have been made in the literature to define the term *smart home*, for example, in [54.11] the term is defined as the “the latest expression of the various ways in which technology in the home has developed”. In [54.12], the notion of a *smart home* is defined more explicitly as “a residence equipped with computing and information technology which anticipates and responds to the needs of the occupants, working to promote their comfort, convenience, security and entertainment through management of technology within the home and connections to the world beyond”. Smart homes typically comprise elements such as network of sensors and actuators, and also entire robotic systems. Current smart home technology includes video monitoring, motion detectors, fall detectors, pressure mats, environment control, health monitoring such as blood pressure, pulse rate, body temperature, weight, and human computer

interaction (**HCI**) technology, for example, to recognize gesture.

The development of smart homes requires a number of technical questions and challenges to be addressed [54.11]: how to convert current home structures and architectures into a smart home, how to standardize smart home components, for example, sensor networks, how to keep the equipment cost at a reasonable level, and how to deal with security and privacy issues.

In the following sections we will describe a number of prominent smart home developments, the Japan Electronics and Information Technology Industries Association (JEITA) house, the *Aware Home* at Georgia Institute of Technology, the *Gator Tech Smart House* at the University of Florida, the *Robotic Room* at the University of Tokyo, the sensorized environment for life (**SELF**) at AIST (the Japan National Institute of

Advanced Industrial Science and Technology). These developments show that current smart home technology goes significantly beyond existing home automation. Accounting for the significant increase in elderly population in the near future many of today's smart home developments pay special attention to improving the quality of life of elderly people. The descriptions below will also address the question on how to integrate robotics systems into smart home concepts.

JEITA House (JEITA, Japan). The Japan Electronics and Information Technology Industries Association (JEITA) house project was carried out from 1999 to 2001 [54.13]. It was implemented in a two-storey Japanese house by several companies that participated in the project. The features of the JEITA house include a keyless system which unlocks the door with fingerprint scanners for personal identification. Based on the fingerprint data, a robot dog AIBO (SONY) would greet the persons entering the house. Functions such as opening curtains, turning on lights, or tuning on the air conditioner could be remotely controlled. The plants could be watered and pets could be fed using cellular phones as remote control devices. This would allow the residents to stay away from the house over longer periods. The components of the JEITA house had their own private IP addresses and were connected to a network.

The JEITA house was equipped with health monitoring sensors for elderly family members. For example, if the system detected something unusual, such as an irregular heartbeat, it sent a message to other members of the family through their cellular phones. Such health monitoring functions are extremely important when an elderly person lives independently. The house could also adapt to the habits of family members. This was an important and interesting novelty, since typically the situation is the other way around: the occupants have to adapt to the system. Lastly, the JEITA house also demonstrated the possibility of integrating robotic systems into homes.

Gator Tech Smart House (Univ. of Florida, USA). The Gator Tech Smart House was built at the University of Florida [54.14]. It addresses the needs of elderly people to live independently and maintain dignity and quality of life at a high age. The house is equipped with many *smart devices* such as smart floors tracking the motion of the occupants of the house, smart blinds automatically adjusting ambient light, smart display, smart cameras, smart phone that can act as remote control to other appliances, location tracking, smart leak detectors, or smart

beds. The exterior of the house has a smart mailbox which alerts residence if mail is delivered and a smart front door which can sense home owners using an **RFID** tag, which allows keyless entry to home owners.

The kitchen of the Gator Tech Smart House includes a smart microwave which uses **RFID** on food packages. This allows the microwave to adjust the settings for cooking the meal. It also informs the resident about the readiness of the meal. The kitchen further comprises a smart refrigerator that monitors food availability and consumption, and detects expired food items. The smart refrigerator can create shopping lists automatically and has an integrated meal preparation advisor based on items in the refrigerator and pantry.

The implementation of such a complex system has raised a number of technical issues and questions, including the development of the smart devices, data handling of networks of sensors, or interconnecting smart devices to other devices in the environment. These questions led to some new research tracks on smart houses, primarily grouped into pervasive computing and mobile computing network research.

Aware Home (Georgia Institute of Technology, USA). Aware Home is a living laboratory for research in ubiquitous computing for everyday activities. This project is conducted at Georgia Institute of Technology [54.15]. The major objective of the Aware Home project is to build an environment that is capable of being aware and keeping track of the states and activities of its inhabitants. Aware Home creates a partnership between human and surrounding sensing and computing technologies. This opens several fields of research not only from the technology point of view but also in terms of the social aspects of the inhabitants. The main research agenda of Aware Home spans human-centered



Fig. 54.32 Georgia Institute of Technology: Aware Home

and technology-centered research, software engineering, and social implications.

Technology and application-centered research focuses on sensor networks, distributed computing, context awareness and ubiquitous sensing, individual interaction with the home, smart floors, or finding lost objects. Research on context awareness is inspired by the fact that humans communicate with each other very successfully by referring to what is called shared context. For communication between humans and computer system, this shared context must be made explicitly. Sensor systems which facilitate the extraction of context need to be developed.

The human-centered research focus on support for the elderly and other social issues. A key concept in supporting the elderly is *aging in place*. Aware Home is designed to support the elderly and allow them to be independent instead of moving them to elderly care facilities. Supporting the elderly leads to the study on cognitive support such as to remind elderly when to take their medicines, guide them whenever they are lost, and locate lost items.

Robotic Room (University of Tokyo, Japan). We have discussed some implementations of Smart Homes. So far, helping the elderly to improve their quality of life is a common goal and this was done in a partly passive way. The term passive implies that there is no physical contact to support the elderly. On the other hand, active support of the elderly requires systems that can physically interact with the elderly and can be augmented to a smart home. The system can also be controlled and can communicate with other smart home devices. With this system, the goal of independent living will be further enhanced. Examples of this system include a robotic partner, robotic walking support system, etc. The discussion below will focus on the goals of and research on such a system, which is an environment-type robot system and which has some elements of a robotic system that can physically interact with humans. This system is referred to as a *robotic room*.

The *Robotic Room* project is being conducted at the University of Tokyo [54.16]. The key concept of the Robotic Room is an environment-type robot system which will provide service to humans. In contrast to industrial robotic systems, which interact with objects, the robotic system in the Robotic Room interacts with humans. There are several target implementations of the Robotic Room, which are more user oriented. One illustration of the Robotic Room is shown in Fig. 54.33,



Fig. 54.33 University of Tokyo: the Robotic Room concept

which shows a robot arm mounted on the ceiling to serve a sick person.

This implementation of the Robotic Room has been considered for a robotic sickroom and one area of research that has been studied is behavior media research, which considers behavior as a communication media between human and robot system. The study of behavior media is further subdivided into behavior measurement and recognition, and behavior expression. An example of behavior measurement is the use of cameras to monitor breathing. This is an unconstrained behavior measurement. Behavior expression by robotic systems can be conveyed by motion, such as dancing to indicate joy.

The second target implementation of the Robotic Room is as a sensing room. As a robotic system, it can accumulate behavior and this leads to the study of behavior accumulation or behavior content. This accumulated behavior can be used for system decision making. This is motivated by the observation that older people are more experienced and can decide correctly based on a certain situation due to their accumulated experience.

The third Robotic Room implementation is as a distributed actuation room. In this robotic room, behavior adaptation research is studied. The robotic systems are designed to accumulate personal behavior information and behave based on the user's behavior. Behavior adaptation research can be subdivided into environment support and operational support. In environment support, the room contains a robotic kitchen which adjusts its height with respect to the user and a robotic lamp that adjust its position and brightness depending on the user's behavior.

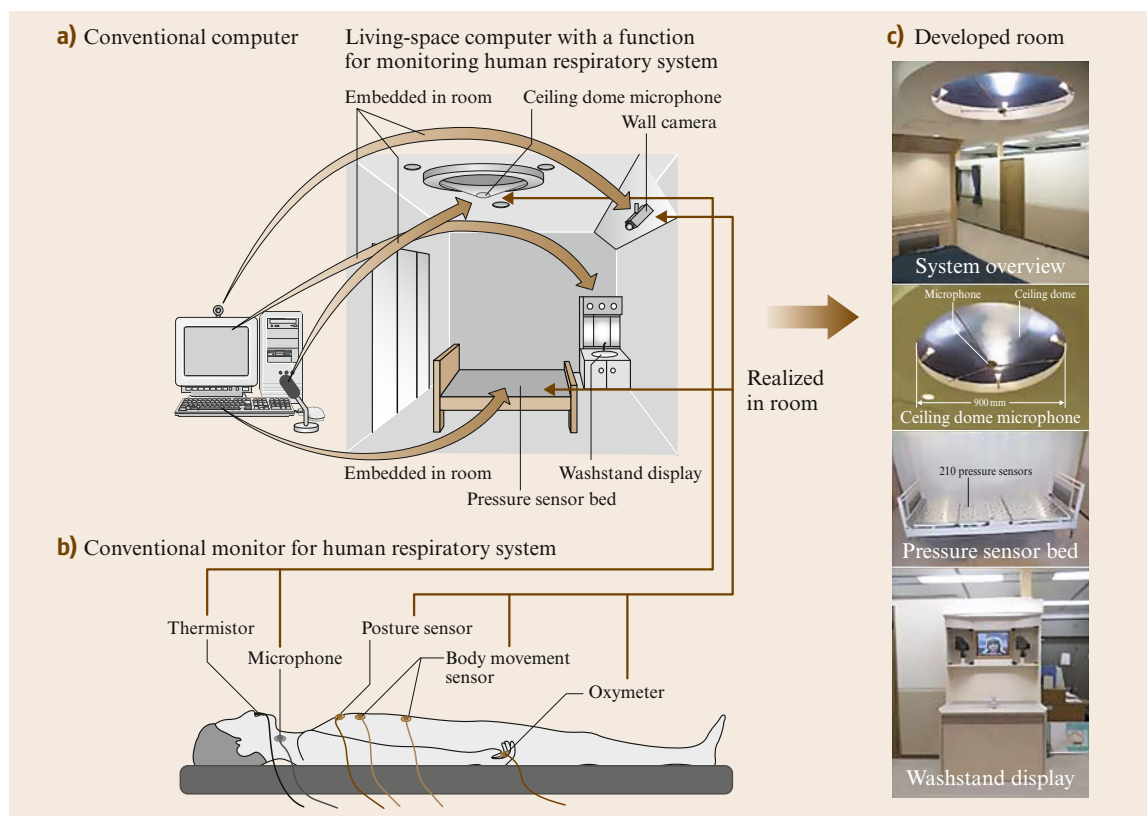


Fig. 54.34 AIST, the sensorized environment for life

SELF – *Sensorized Environment for LiFe* (AIST, Japan). **SELF** stands for Sensorized Environment for LiFe [54.17]. The objectives of **SELF** are to develop a network of sensors which are embedded in the environment, information gathering using the network sensors, storing and analyzing the information, and reporting of useful information to assist and maintain good health. The basic advantages of **SELF** due to the embedded nature of the sensors are: (1) no size limit, weight, and power source, (2) it does not disturb the human, (3) it does not impose physical restrictions, and (4) sensors are rarely broken since they are fixed to the environment.

SELF can be viewed as a system that monitors a person's behavior or activity and represents the data objectively in an approach known as self-externalization. The motivation of **SELF** is that humans sometimes cannot notice a change in their condition which greatly

affects their health status without a medical doctor. Therefore, the use of network sensors to monitor human behavior and report useful information that greatly affects the health status will further improve quality of life.

The **SELF** study considers behavior as a means of communication and sensors embedded into the environment as one way to observe a person's behavior. The **SELF** implementation is shown in Fig. 54.34, which shows a bed with a sensor, a ceiling with microphones, a washstand with a display, etc. The bed with sensors can determine the time the subject sleeps and wakes up, their posture during sleeping, and their breathing pattern. The microphones attached to the ceiling can detect snoring or normal breathing sounds. Based on the monitored data, the washstand display is used as an output device to provide the subject's health status and thus create feedback to the subject.

54.5 Domestic Robotics: It Is the Business Case Which Matters

After having seen the plethora of domestic robots in the previous sections, we are tempted to say that domestic robotics are no longer a dream or a vision, they are a reality. More than two million domestic cleaning robots sold within four years appears to be a tremendous success story. Of course, as we stated earlier, this is certainly a milestone in the development of domestic robotics and service robotics in general. It is a success story since Roomba created a market. Compared to the sales figures for industrial robots, which are coarsely estimated to be around 100 000 units per year [54.18], the figures for domestic cleaning robots are five times as high.

However, the figures themselves present a naive fallacy. The estimated annual market volume for industrial robots totals around 3 billion EUR. The annual market volume for domestic cleaning robots, given the published figures, is estimated to be around 0.15 billion EUR, which is 20 times smaller. Furthermore, the market size for full-size household vacuum cleaners in the US was estimated to be close to 20 million units in 2003 [54.19]. So the ratio between domestic cleaning robots sold worldwide from 2002 to 2006 and regular vacuum cleaners sold only in the USA during the same time is 1:40. Conservatively estimated, this ratio may be around 1:400 or worse worldwide. Given these figures domestic cleaning robots still seem to be considered to be gadgets rather than appliances.

Is that a reason for joy? Of course, it is. The growth potential is enormous. If only 2% of all domestic vacuum cleaners sold per year were robots this would mean a growth of 800%. Having realized that, shall we now all invest in domestic cleaning robot manufacturers and then order a Ferrari? After all, a growth rate of 800% is an impressive target. As the reader might guess there is a problem here. Housewives (and househusband) are like companies and business people. They ask for a return on their investment. They invest in new equipment only if there is a return on investment and only if the new equipment has a competitive price-to-performance ratio. If it is cheaper to hire a cleaning person than to buy a cleaning robot, why should one buy a cleaning robot. The same holds if the device is cheap but does not do the job, as you still need to do the cleaning yourself.

Apparently, it all boils down to the question, how much is the average housewife or househusband ready to pay for a cleaning robot and what do they expect and how much are they ready to pay for a gadget? Are

they ready to pay 3500 USD for a device which at least promises to systematically clean their living room or do they prefer to pay 100 USD for a toy which gets frequently trapped by cords and cables and breaks after a few month and create more noise than cleanliness?

Let us briefly look into professional cleaning and see whether the situation is different there. The professional cleaning of 1 m² floor costs on the order of 5–10 EUR cents if done manually with a mop (all figures are qualified estimates which are sufficient for a case study but do not claim to account for a specific application or business case). This includes the cost of labor and material. Now assume a 500 m² entry hall of a public building which is cleaned once a day, five days a week, 52 weeks a year. This totals approximately 6750 EUR per year.

Now assume that the cleaning is done by a robot which is deployed by a professional cleaner or housekeeper. Let us further assume that deploying and maintaining the cleaning robot takes the cleaner 20 min per day and that the cleaner has an hourly wage of 20 EUR. The labor for deploying and maintaining the cleaning robot accordingly costs approximately 1750 EUR per year.

What does this figure mean? Well it means that the cost for the cleaning robot must not exceed 5000 EUR per year, including consumables, repair, and purchase. Assuming a depreciation period of three years and assuming that consumables and repair make up 1000 EUR per year this means that the price for the robot must not exceed 12 000 EUR. Not a single one of the professional cleaning robot developed in the past 20 years has even gotten close to such a price. On average the price for a professional cleaning robot is around 50 000 EUR, four times the price that would satisfy our business case.

It should no longer be surprising that the number of professional cleaning robots installed over the past 20 years totals fewer than 500 units worldwide. What is even more disillusioning is the fact that, out of around 15 developments of robots for professional cleaning in the past 20 years, not a single one has survived and ever made it to a successful product [54.20].

What is the conclusion which we can draw from this? Well, launching a successful product does not only require a technology push but also a market pull. It is not the enthusiasm of the engineers which matters. It is the need of the customers and the business case.

54.6 Conclusions and Further Reading

In this chapter we have tried to give a survey of the state of the art in domestic robotics, smart appliances, and smart homes. We have tried to make this survey as comprehensive as possible, but do not claim that it is complete. In view of the huge market potential – we talk about hundreds of millions of customers – the field is developing rather dynamically. New products and new companies appear and disappear rather frequently.

We have discussed the technical challenges and open problems which have to be faced when developing a robot for a household task. Some of these challenges and problems require significantly more research to generate not only theoretical but also practical solutions; some are engineering problems, which are no less trivial. We have briefly looked into some new technologies which have the potential to boost the field and to make products more robust.

We have argued that whether or not a domestic robot makes its way onto the market is not only a matter of the engineers' ingenuity but also of the customers' needs and expectations and above all of the customers' wallet. There is a business case which needs to be met and not only technology to be developed.

We have given an overview of domestic cleaning robots for floors, windows, and pools and we have presented a selection of lawn-mowing robots. We have further looked into the latest developments in smart appliances such as ironing robots, intelligent refrigerators, and intelligent wardrobes. Some of these smart

appliances are intelligent networked devices but are not service robots with motors, wheels, and sensors. We have finally looked at smart homes, in which domestic robots are or can be embedded.

Concluding this chapter one can say: some of the robots which Karel Capek and his brother Josef may have imagined in 1920 have become everyday reality 90 years later. Others, however, are still waiting to see the light of day. And some may never see.

The publication that blazed the trail for service robotics as a whole and for robots for *household tasks* in particular is *Joe Engelberger's* book *Robotics in Service* [54.21]. This book provides an amazing overview of potential applications of robotic technology for service tasks and, although it is nearly 20 years old, is still as inspiring as ever. A selective survey of the field of service robotics including domestic applications such as domestic cleaning and lawn mowing is given in [54.22]. A historical overview (until 2000) of the development of cleaning robotics from the very first prototypes to the very first products is presented in [54.23]. [54.24] reviews a selection of commercially available domestic robots and discusses open technical issues which have an impact on the development of domestic robots. The economic development of the field of service robotics as a whole branch and of domestic robots such as cleaning robots, robotic lawnmowers, entertainment robots, and assistive robots is documented in *World Robotics*, the annually published statistics of the Statistical Department of the International Federation of Robotics [54.18].

References

- 54.1 M. Schofield: Cleaning robots, *Service Robots Int. J.* **1**(3), 11–16 (1995)
- 54.2 J. Borenstein, H. Everett, L. Feng: *Where am I? Sensors and methods for mobile robot positioning*. University of Michigan. Tech. Report. (1996)
- 54.3 H. Choset: Coverage for robotics: a survey of recent results, *Ann. Math. Artif. Intell.* **31**, 113–126 (2001)
- 54.4 R.V. Bostelman, T.H. Hong, R. Madhavan: Towards AGV safety and navigation advancement obstacle detection using a TOF range camera, 12th Int. Conf. Adv. Robot., ICAR (2005)
- 54.5 R.D. Schraft, M. Hägele, K. Wegener: *Service-Roboter-Visionen* (Hanser Verlag, München 2004)
- 54.6 *SwissRanger SR-3000*. <http://www.swissranger.ch/about.php>
- 54.7 J. Hightower, G. Borriello: Location systems for ubiquitous computing, *IEEE Comput.* **34**(8), 57–66 (2001)
- 54.8 T. Weber: Smart floor: Neue Grundlagen für Robotersteuerung, CeBIT (2006)
- 54.9 E. Prassler, B. Kluge, T. Kämpke, M. Strobel: RFID navigation of service robots on smart floors, 3rd Int. Workshop Adv. Service Robots (2006)
- 54.10 Security firm develops fuel-cell robot. Press Release. <http://www.fuelcellsworld.com/Suppage3152.html>
- 54.11 D. Gann, J. Barlow, T. Venables: *Digital Futures: Making Homes Smarter*. Report published by Chartered Institute of Housing. (1999).
- 54.12 R. Harper: *Inside The Smart Home* (Springer, London 2003)

- 54.13 JEITA. <http://www.eclipse-jp.com/jeita/>
- 54.14 Gator Tech Smart House. <http://www.icta.ufl.edu/gt.htm>
- 54.15 Aware Home Research Initiative. <http://www.awarehome.gatech.edu/>
- 54.16 T. Sato, T. Harada, T. Mori: Environment-type robot system *Robotic Room* featured by behavior media, behavior contents, and behavior adaptation, IEEE/ASME Trans. Mechatron. **9**(3), 529–534 (2004)
- 54.17 T. Hori, Y. Nishida, T. Suehiro, S. Hirai: SELF-Network: design and implementation of network for distributed embedded sensors, IEEE/RSJ Int. Conf. Intell. Robots Syst. IROS (2000)
- 54.18 World Robotics: Statistics, Market Analysis, Forecasts, Case Studies and Profitability of Robot Investment. Int. Federation of Robotics. (annual)
- 54.19 R. Kirkpatrick: 2003 Association forecasts: vacuum cleaner manufacturers association, Appliance Magaz. (2003)
- 54.20 E. Prassler: Human friendly service robots: pitfalls, potentials, perspectives Invited talk, 15th German-Japanese Forum IT (2005)
- 54.21 J.F. Engelberger: *Robotics in Service* (Kogan Page, London 1989)
- 54.22 R.D. Schraft, G. Schmierer: *Service Robots: Products, Scenarios, Visions* (AK Peters, Natick 2000)
- 54.23 E. Prassler, A. Ritter, P. Fiorini, C. Schaeffer: A short history of cleaning robots, Auton. Robots **9**(16), 211–226 (2000)
- 54.24 H.I. Christenen: Intelligent home appliances, Proc. Robot. Res. Tenth Int. Symp. (2001)

Robots for Education

David P. Miller, Illah R. Nourbakhsh, Roland Siegwart

This chapter provides an overview of the key ingredients that make successful education robots possible. Two very popular outlets for public interaction with robots are the robot tournament and the informal learning venue (e.g., the science museum). Section 55.2 provides a survey of the very popular world of robot-themed tournaments, which have already impacted tens of thousands of students across diverse geographic and age group boundaries [55.1–5]. Section 55.5 provides an overview of robotic installations in informal learning spaces. Robotic technology has now proven to have sufficient robustness and engagement to be a principal component of interactive exhibitry for a new generation of hands-on, active-learning museums [55.6].

To make interactive, educational robots successful, a new level of technology robustness and standardization is required, and significant progress has been made on this front in the past decade. Educational robot devices consist of both hardware (preassembled or as kits or components) and software (both as source code and programming environments). Section 55.3 discusses physical robot platforms that have achieved notable success, while Sect. 55.4 describes both low-level controllers that interface those platforms to high-level computation, as well as the top-level programming environments themselves.

Finally, an important class of tool in the study and execution of educational robotic systems is the ability to evaluate the efficacy of a robot system formally in an educational context. Numerous tools from human-computer interaction, cognitive psychology, and education have demonstrated their usefulness in this regard. Section 55.6 sum-

55.1 The Role of Robots in Education	1284
55.1.1 Interest in Science and Engineering	1284
55.1.2 Teamwork	1284
55.1.3 Problem Solving	1284
55.2 Educational Robot Tournaments	1285
55.2.1 Origins	1285
55.2.2 A Taxonomy of Robot Tournaments	1285
55.2.3 The Entertainment Link	1286
55.2.4 Tournaments for Education	1287
55.3 Education Robot Platforms	1287
55.4 Education Robot Controllers and Programming Environments	1290
55.4.1 Robot Controllers	1290
55.4.2 Edutainment Programming Environments	1292
55.5 Robots and Informal Learning Venues (Museums)	1292
55.5.1 Tour-Guide Robot Examples	1292
55.5.2 Lessons Learned	1295
55.6 Educational Evaluation of Robot Programs	1296
55.6.1 Design-Time Assessment	1296
55.6.2 Formative and Summative Evaluation	1296
55.7 Conclusions and Further Reading	1298
References	1298

marizes the manner in which conventional analytical tools may be used to evaluate unconventional educational programs that tap robotic technologies as learning tools across a variety of ages and in both formal and informal learning venues.

In recent years robots have penetrated the education market, both as motivational tools for research programmes and as concrete, real-world deployments

that demonstrate the state of the art in robotic technologies. The newest education robots, and even entertainment robots to some degree, share the no-

table feature of human–robot interaction with the general public. In contrast to a space exploration robot or a nuclear cleanup robot, these newest, interactive robots do not function solely with the well-trained

specialist; they interact with individuals and groups with the goal of inspiring learning, providing engaging recreation, and even providing therapeutic value.

55.1 The Role of Robots in Education

Education is itself a broad area of application for robotics, with a rich history that started at least with notions of active learning disseminated by Seymour Papert and others [55.7]. Karel the Robot still serves as a metaphor for concrete activity in numerous introductory programming courses. Karel the Robot is merely an on-screen simulation of a two-dimensional robotic world. This first of three roles that robots can play in the educational setting is the *robot as a programming project*. The robotic system is a focus of problem set assignments, a black box to be programmed in order to create a concrete physical manifestation of the art of computer programming. Numerous studies have shown that poor engagement and retention statistics in courses such as Introduction to Programming (CS1) often stem from students' inability to see how the skills they learn can have a concrete impact on what they care about: their physical world, their friends, and family. Physical robot programming projects can cast coding problems into the real world, making those skills push back on one's world, and thereby achieve a level of significance and engagement that, for instance, computation of the Fibonacci sequence on a computer screen cannot inspire [55.8–16].

The second role for educational robotics is that of the *robot as a learning focus*. Mechatronics-style courses can focus on the creation and use of a physical robot as a goal in and of itself. Robots do this by stimulating general interest in science, technology, and engineering, and thereby show students that they can have an active role in shaping technology in their future. Such an application-oriented class has great benefits in cases where students learn best through integrative, project-oriented learning, and such classes have demonstrated significant lifelong-learning results in areas such as teamwork, problem-solving, and self-identification with technology-focused careers [55.7, 17–21]. In the United States, the National Academy of Engineering has identified trends reporting dropping technology literacy as troubling phenomena throughout the nation. Programs that reconnect students with complex mechatronic systems, providing the self-empowerment for lifelong, active relationships to technology artifacts, serve as one possible avenue to stem this dangerous trend away from

knowledge about the technology artifacts on which society depends so completely. Demonstrated learning themes in educational analyses include the following.

55.1.1 Interest in Science and Engineering

The study of robotics can be sufficiently rewarding, intellectually and emotionally, as to generate general enthusiasm for the follow-on study of science and engineering. Some educational robotics programs have shown promise in increasing retention rates among female students [55.22, 23], who are underrepresented in technology-focused fields.

55.1.2 Teamwork

An important result of team-based project-based robotic building experiences is that those with low self-esteem in their technology capability, teamwork skills, and communication skills can find measured improvement, leading to mitigation of fear towards working in teams on high-technology ventures [55.19, 21].

55.1.3 Problem Solving

The study of a systems engineering phenomena – a complex robot system – provides lessons in hierarchy, risk mitigation, planning and diagnosis that far outweighs the benefits of silo-driven, single-topic explorations of debugging and problem analysis. The resulting problem solving and decomposition skills have lifelong applicability to any complex system, see e.g., [55.7].

The third role for educational robots is that of the *robot as a learning collaborator*. In this case students are not designing robots, but rather conducting inquiry during which a high-functioning robot can serve as an all-season companion, aide, and even intellectual foil [55.24, 25]. Robots that have social capabilities are well suited in this regard, and this is an area where the potential for application greatly outweighs what has, so far, been achieved. There are two reasons why such a social role for robots in education holds particular promise. First, because of their novelty, robots can in-

spire dedication in learning, just as is possible in the case of the above two roles. Second, students have few preconceived expectations for the behavior of robots. In therapeutic cases, such as students with autism spectrum disorder (ASD), the robot can play an important role, as studies have shown that those students place dispro-

portionate interest in mechanical systems [55.24, 26]. In all cases the interaction can itself be a form of discovery and engagement quite different from traditional, more staid educational methods, and thus this can even border on the application area of entertainment robotics.

55.2 Educational Robot Tournaments

Competition catches people's attention and makes them want to get involved. This is true in almost all fields of endeavor. In education, competitions are commonplace from spelling bees to science fairs. Over the past 20 years robotic competitions have also come into their own; garnering the level of attention from students, teachers, and parents previously found almost exclusively in more traditional sports.

In this section we provide a brief history of robot tournaments, look at some of the popular current tournaments, and overview how they are being used to supplement standard educational curricula.

55.2.1 Origins

In 1970 professor H.H. Richardson adopted MIT's Introduction to Design Course (course 2.70), in the Mechanical Engineering (ME) department, and transformed it into a design contest. He had the students work in teams to design and create machines that would be operated by the students to compete against one another. As the course evolved over the years, and under new instructors, it became both popular and famous.

Several students in the Electrical Engineering and Computer Science department at MIT, envious of the ME course decided to create their own, more autonomous version of both the course and the contest. Since 1987, MIT students have held the 6.270 course during the intercession between the fall and spring semesters. While initially simulation, by 1991 the contest involved untethered autonomous robots in a head-to-head elimination tournament [55.27]. While only open to MIT students and, with the exception of the occasional documentary, poorly publicized, these two courses have become the models for many teleoperated machine contests and robot contests around the world.

55.2.2 A Taxonomy of Robot Tournaments

There are many possible ways to distinguish the multitude of activities often lumped together as robot contests.

For our purposes here we will use the following qualities.

Autonomy

The level of autonomous action that robots exhibit during a tournament varies widely. Some tournaments are completely radio controlled (e.g., *BEST*TM [55.28]) with no independent actions or decisions being undertaken by the 'robots' in the tournament. In other contests, the machines are teleoperated – still under real-time control by their operators, but doing some processing on the commands, and sometimes mixing in feedback from onboard sensors (e.g., *FIRST*[®] [55.5]). Autonomous robot contests have little or no interaction by the robot operators, but even in completely autonomous robot contests there is a spectrum. Some autonomous robots simply replay a canned set of actions – operating completely open loop. Others control all of their actions based on feedback from onboard sensors, and still others *learn* as they go, modifying not only their actions, but their internal programming as well. One generalization that can be made with regards to robot tournaments and education is that a contest that is more autonomous usually has more educational content in the areas of software/programming, while contests with little or no autonomy emphasize the mechanical design and physical implementation aspects of robotics.

Performance Versus Opposition

Some robotics tournaments are based on ranking the absolute performance of the robot against the course. Other tournaments have contestants ranked based on their triumph over a series of specific opponents. In the latter case, it does not matter how well the victorious robot performs in a given round, as long as it performs better than its opponent for that round. Trinity College's *Firefighting Robot Contest* [55.29] (TCFFHRC) ranks teams by how quickly they can carry out the task (locating and extinguishing a candle) and is an example of a performance-based tournament. The scoring in *RoboCup Junior Dance* [55.3] is more subjective, but is

still based on individual performance against an absolute scale. Sumo contests such as [55.30] are opposition-based tournaments sometimes known as head-to-head. Performance-based contests allow robot designers to implement more intricate strategies, since the environment is more predictable than if there were an opposing robot running amok through the course. In oppositional tournaments, robots must be designed to cope with the actions of their opponents. Both types of tournaments have their merits and each allows detailed problem solving. In theory, most people find head-to-head tournaments more exciting and entertaining. However, as with traditional human sumo, head-to-head tournaments often involve head-to-head collisions followed by lengthy, and less exciting, periods of robots trying to push or disentangle themselves from their opponents.

Heritage: Fixed Game
Versus New Game Every Year

Is it more educational for students to examine how others have tried to solve a problem and then build on that to create their own incremental solution? Or is it better for students to tackle something new, without extensive examination of precedent? The educational community does not have a definitive answer, but different robot tournaments can be used to support the educational methodology of choice. Tournaments such as *Robocup* [55.32] are prime example of tournaments with a long heritage. In addition to the game and rules being well known (derived from soccer) and essentially remaining unchanged from year to year, there are a number of annual publications (e.g., [55.33]) that detail successful robots and techniques used in that year’s contests. *Robocup* is a study in steady incremental improvement. Other contests, such as the *Botball*[®] robot tournament, use a different game every year. One of the motivations for doing this is so that established teams obtain the game rules at the same time as new teams – so whether experienced or not, everyone has to build

a new system for the new task, and has the same amount of time to do so [55.4]. As with all of these dimensions, there is no clear advantage of one methodology over the other. Some educators prefer to have a new activity every year to help keep themselves and their students engaged. Others prefer to have the same game every year, with the accompanying wealth of lessons learned and how-to guides that process makes available.

Table 55.1 presents a number of established robot tournaments using this taxonomy, although this is not a complete listing. Readers interested in finding out more information about a particular tournament should consult the cited website. All of the tournaments below are for students, though the age eligibility and level of adult involvement varies widely between the events.

55.2.3 The Entertainment Link

Tournaments are used (rather than a course assignment or demonstration) in part because of their entertainment value. A tournament is hopefully engaging to both the participants and the audience. However, the entertainment value of a tournament may not be linked to the technical sophistication of the robots. *Battlebots*[®] and its variations have been very successful as mass market entertainment. These robots are typically radio controlled, and while sometimes mechanically sophisticated, they are computationally simplistic. The Association for the Advancement of Artificial Intelligence (AAAI) *mobile robot contest* [55.34] usually has very computationally sophisticated robots – however that sophistication usually has an accompanying decrease in audience entertainment value. AAAI robots strive to do what everyday humans easily do – go to specific places, manipulate household objects, give talks, and answer questions, but the robots tend to be much slower than humans performing these activities. *Battlebots* do what humans are not allowed to do — use power tools and blunt instruments to demolish their op-

Table 55.1 Examples of different types of robot tournaments

Tournament	Web	Autonomy	Contest type	Heritage	Notes
BEST	[55.28]	Radio control (R/C)	Head to Head (HtH)	New	
Botball	[55.1]	Autonomous	HtH + Points	New	Multiple robots per entry
FIRST	[55.5]	Teleop	HtH	New	Students teamed with engineers
Micro Maze	[55.31]	R/C to auton.	Time.	Fixed	Cubic cm robots
Sumo	[55.30]	R/C to auton.	HtH	Fixed	Divisions based on weight & autonomy
RoboCup	[55.32]	Autonomous	HtH	Fixed	Teams of robots
RC Jr. Dance	[55.3]	Autonomous	Points	N/A	Very free form
TCFFHRC	[55.29]	Autonomous	Time	Fixed	Random placement of fire



Fig. 55.1 Middle-school students present their paper at the National Conference on Educational Robotics (NCER)/Botball conference and then have their autonomous machine-vision-guided robots score points in the tournament

ponents. In robot tournaments, as in the movie business, violence is often more entertaining than demonstrations of intellectual ability and human activity.

55.2.4 Tournaments for Education

Professional organizations such as the [AAAI](#), Institute of Electrical and Electronics Engineers ([IEEE](#)), and others often hold tournaments for their student members at their major conferences. For these organizations, the purpose of the tournament is to attract students to attend the conference and become active in the organizations. The activity of building the robot also sharpens the students' skills and gets them excited about technologies and businesses linked to that organization and motivates them to pursue relevant degrees [55.35].

Many universities and colleges use robot tournaments as part of their curricula. Some create class tournaments (e.g., MIT 6.270) or use standard games such as *Beyond Botball* [55.36] that are distributed

for that purpose. Others use some of the tournaments mentioned in Table 55.1 as projects for their curricula [55.37].

However, the bulk of robot tournaments are aimed at middle- and high-school students. Here the motivation is not so much to teach detailed skills as to spark creativity while at the same time emphasizing basic concepts and an appreciation for the fundamentals of science, technology, education, and mathematics [55.38]. Some tournaments have added activities such as documentation, presentation, reports, and even tests [55.39] to reinforce and expand upon these fundamentals (Fig. 55.1). To assist educators in guiding the students, many tournaments have curriculum materials linked to their activities. National Aeronautics and Space Administration ([NASA](#)) has collected links to many of these materials and made them available on the Internet [55.40]. Evaluating the effectiveness of these materials and programs in meeting their educational goals is the subject of Sect. 55.6.

55.3 Education Robot Platforms

Education robot platforms fall into one of two main categories. The first is the research platform. These expensive, precision robot platforms were built to support research in robotics and related disciplines, or as research projects in their own right. Many of the humanoid robots discussed elsewhere in these volumes are examples of this. These robots, such as Honda's *Asimo*, are used for advanced research as well as for entertainment.

Other less sophisticated but still precise and expensive platforms are used as the basis for the museum robots discussed in Sect. 55.5.

A second class of education platforms (and the ones to be discussed in this section) are lower-cost robot platforms, or platform kits. These platforms are inexpensive, exist in large numbers, and are widely used in schools and by hobbyists.



Fig. 55.2 (a) Hero-1, (b) Hemisson, and (c) Amigobot

Education robots really got their start in the early 1980s with the introduction of the Heathkit *Hero-1* (Fig. 55.2a). Hero robots were sold as kits, both to keep the price down and to encourage users to learn how robots were made. However, while detailed assembly instructions were included, no information on the theory or principles behind the assembly were given; the kits, by themselves, did not provide a satisfactory educational experience. Unfortunately, neither the Hero robots, nor several other similarly sized and priced personal robots that came out in the 1980s, reached a level of capability and ease of use to attract an economically viable customer base. Heathkit and the other companies have since gone out of business.

In the intervening years several other companies have tried to make personal robot systems. Many of these companies started with more capable research robots. With few exceptions, these companies have faded away or gone into other markets. Most of the companies now mass marketing education robots do so as a side business. The primary business for most of these companies is consumer electronics, toys, and with one noticeable

exception consumer robots that were supposed to be neither educational nor entertaining.

K-Team [55.41] is one of the few companies making robot platforms specifically for the education market. The *Hemisson* (Fig. 55.2b) is a low-cost, less compact version of K-Team's Khepera research robot. It has reduced computational power and few sensors when compared to most research robots, but is priced and designed for robotics classes in secondary and university classes. Activemedia is the other research robot company that produces lower-cost educational robots. Their *Amigobot* platform is shown in Fig. 55.2c.

The LEGO® Mindstorms RCX was the first robotic platform from the LEGO company, and has had broad appeal across many demographics. The LEGO computation systems are discussed in Sect. 55.4. However the LEGO blocks themselves have been a mechanical prototyping system for robots long before LEGO ever introduced their own controllers. The LEGO bricks make RCX a good tool for early and fast robotic explorations. The newly released LEGO robot system, Mindstorms NXT (Fig. 55.3a) has a new processor and some new sen-

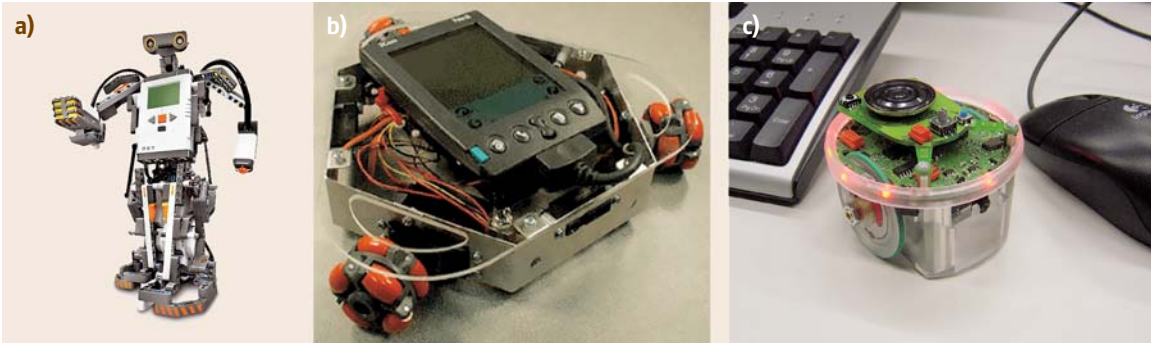


Fig. 55.3 (a) LEGO NXT, (b) PPRK, and (c) e-puck robots



Fig. 55.4 (a) Garcia, (b) ER1, and (c) KHR-1 robots

sors, but relies on the popular LEGO Technics building system for all of the mechanics.

The Palm Pilot Robot Kit (PPRK) platform [55.42] (Fig. 55.3b) is a low-cost kit combining a mobile base and a personal digital assistant (PDA), originally a PALM Pilot. The PALM provides computational power and a graphical user interface for robot control, and communicates with lower-level robot driver circuitry via the PALM's serial port. The result is a very compact omnidirectional platform with three distance sensors (in its basic configuration) that constitutes a starter kit for those who want to begin by building their own robot from kit parts.

E-puck [55.43] (Fig. 55.3c) is the first desktop robot (70 mm in diameter) based on an open-hardware concept. Equipped in its basic version with a diverse sensor package (three-dimensional accelerometer, proximity sensors, three microphones, color camera), several actuators (stepper motors, speakers and numerous LEDs), Bluetooth[®] communication and a good processor capable of signal processing (dsPIC family), it is relatively inexpensive due to its simple mechanics. It is a desirable

tool for education in technical schools and universities.

The Garcia robot from Acroname [55.44] (Fig. 55.4a) is a small-size platform (25 cm × 18 cm × 10 cm) that can be extended with a XScale-based board. It is sufficiently small to develop experiments in small environments and yet this robot is sufficiently large for diverse payloads, including a small manipulator mechanism and an array of sensors.

The ER1 robot from Evolution Robotics [55.45] (Fig. 55.4b) constitutes a simple extruded aluminum frame kit supporting a laptop and equipped with stepper motors and wheels. This robot focuses interaction on the onboard laptop, enabling onboard computation as well as onboard vision via a universal serial bus (USB)-based camera. This robot system, which is going out of production, comes with a sophisticated software environment for navigation and visual recognition of artifacts in the robot's field of view.

The KHR-1 [55.46] (Fig. 55.4a) is an initial humanoid robot at an affordable price, based on a set of 17 servomotors. The growing family of humanoid robots



Fig. 55.5 (a) Robosapien, (b) Roomba, and (c) Create robots

still depends upon large numbers of servo actuators and very few sensors; however, due to the draw of humanoid robots, these kits are nonetheless excellent motivational tools.

Robsapien [55.47] is a popular toy humanoid robot (Fig. 55.5b). Because this is an infrared (IR)-controlled toy, using the IR link from a computer or PDA is relatively straightforward. Some hobbyists have hacked deeper into the system so that the Robosapien can be used as a fully fledged computer controllable robot platform.

The iRobot® *Roomba*® [55.48] (Fig. 55.5c) is a mass-produced, inexpensive household vacuum robot (33 cm × 33 cm × 7 cm). With millions sold, the Roomba is currently the most common robot in the world. The Roomba was not initially meant to be an education robot – but rather a service robot doing dirty and dull work. However, much like the Robosapien, the robot hobbyists quickly started using the Roomba

as a platform, and starting in 2005, iRobot responded to requests and included an exposed serial port and released an application programming interface (API) for controlling the Roomba and reading its sensors from an external computer [55.49]. A community of Roomba hackers has come into existence using Roombas for everything from musical instrument digital interface (MIDI) devices to security robots [55.50]. Many schools are adopting the Roomba, combined with one of the controllers described in Sect. 55.4.1, as a platform for classes and experimentation. A new product, the iRobot *Create*™ was released in 2007. This product is a Roomba without the vacuum or brushes, but with additional interfaces to encourage experimentation. The iRobot Create can be purchased with a processor of its own or easily interfaced with robot processors such as the XBC [55.51] or small low-power general-purpose processors such as the *Gumstix*™ [55.52, 53].

55.4 Education Robot Controllers and Programming Environments

Education and entertainment activities in robotics sometimes run at cross purposes. For entertainment we want robots that are active and very easy to use and to control. Products like the LEGO Mindstorms provide both a robot controller and a programming environment that meets those criteria.

However, when it comes to robot education, it is important to remember that in most cases we are using robots to inspire and motivate students to learn many things, of which robotics is just one. Robotics educators, especially at the K-12 (students aged approximately 5–18 years level, are hoping to teach students general science, programming, math, and engineering techniques. The toy programming environments used in some products such as Mindstorms may be inappropriate for this mission.

This section overviews some of the popular robot controllers and programming environments. Usually the two are linked, but there are some environments that work with several different controllers and vice versa.

As with the robot tournaments described in Sect. 55.2, many of the robot controllers and their programming environments in use for tournaments and by hobbyists can trace their origins back to the MIT 6.270 course [55.27]. At an even more fundamental level, the origin of most robot controller starts with the Motorola 68HC11 processor.

55.4.1 Robot Controllers

The venerable HC11 is a single-chip processor with small amounts of onboard flash and random-access memory (RAM) memory. It also has a number of input/outputs (I/O) and timing ports that make it relatively easy to interface to sensors and external devices. The 8-bit HC11 uses the same instruction set as the Motorola 6800 series of processors, which have been in existence in one form or another for more than three decades.

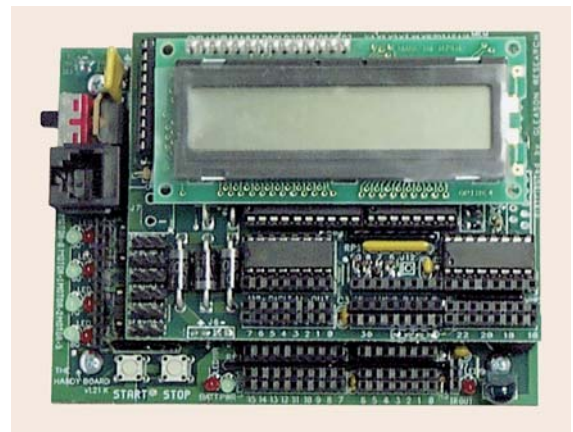


Fig. 55.6 The HandyBoard with its expansion board

The HC11 was used by Anita Flynn and Joe Jones in a board for the MIT robot talent show in 1991, which spawned the Book Board [55.54], the MIT 6.270 board, and Fred Martin's Handyboard. The last of these became the de facto high-end robot controller for hobbyists and autonomous robot contests for most of the 1990s.

The Handyboard (Fig. 55.6) can provide pulse-width modulation (PWM) control for four direct-current (DC) motors, read eight digital and eight analog sensors, control a couple of light-emitting diodes (LEDs) and has a 32-character display. With the expansion board, eight lines of digital input-output (DIO) and control for six R/C servos is also added. The Handyboard can be programmed using 6811 assembly language and several other languages including Java and C.

The other low-cost commercial robot controller in widespread use throughout the 1990s was the BASIC Stamp [55.55] (Fig. 55.7). This programmable interrupt controller computer (PIC)-based controller could be programmed in a special (often assembly language-like) version of the BASIC language. While very small in memory, and usually requiring supplementary electronics for most sensors and actuators, the Stamp was very popular because of its low cost and flexibility.

In the late 1990s, spurred on in large part by the growing interest in robot tournaments, a renaissance in robot controller hardware started making available a number of controllers ranging from advanced PICs to full-blown personal computers (PCs). A sampling of some of the controllers now available is presented in Fig. 55.8

The RCX (Fig. 55.7) and its successor, the soon to be released NXT (Fig. 55.3a), are widely available controllers from LEGO. These are very easy to interface to the special sensors and motors produced by LEGO, using variations of LEGO's brick interface. However, this simplicity of use makes it more difficult to connect to other sensors and motors. These foolproof interfaces also take up a lot of physical space, limiting the controllers to just a few sensors and actuators. While the central processing units (CPUs) are relatively powerful, most of this horsepower remains unused because of the limitations of the interface. However, these controllers are not very expensive, are mass produced, and require absolutely no knowledge of electronics, mechanics (or when using LEGO's software) programming in order to get them to control simple robots.

For the more technically inclined hobbyist or student, but one on a stricter budget, there are the PIC-based controllers. The OOPic system [55.56] (Fig. 55.8), uses an object-oriented programming system, and includes

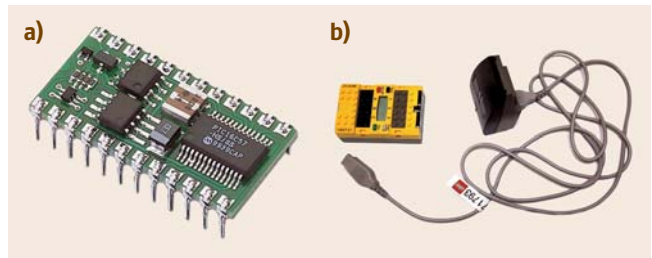


Fig. 55.7 (a) The BASIC Stamp 2 and (b) the LEGO RCX

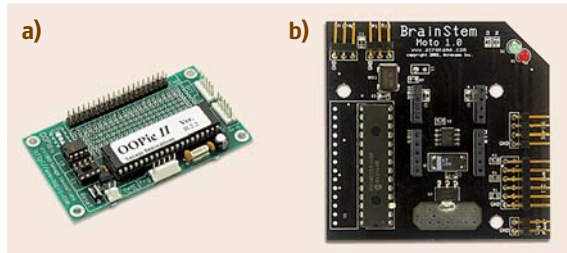


Fig. 55.8 (a) The OOPic Controller and (b) the Brainstem MotoBoard

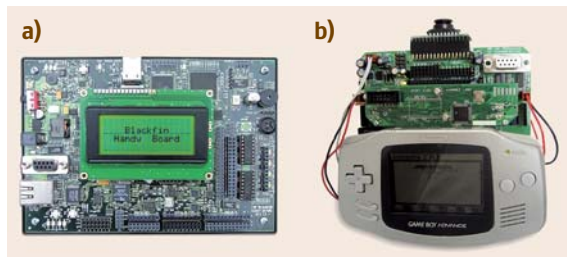


Fig. 55.9 (a) The Black Fin Handy board and (b) the Xport Botball Controller (XBC) with camera

much, but not all, of the interface electronics needed to get a robot up and going.

Acroname's Brainstem system [55.57] (Fig. 55.8) uses a small reduced-instruction-set computer (RISC) processor to create a low-cost modular controller. Modules are linked over an inter-integrated-circuit (IIC) bus. The Brainstem is programmed in the tiny embedded application (TEA) language, a variant subset of the C language.

For more capability including proportional-integral-derivative (PID) motor control and vision processing there are controllers such as KIPR and Charmed Labs, XBC [55.58] (Fig. 55.9), and the Blackfin Handy Board [55.59] (Fig. 55.9). Both of these boards share the same basic architecture of a RISC processor combined with a field-programmable gate array (FPGA). The FPGA comes preprogrammed with a va-

riety of useful functions (e.g., motor control) but can be reconfigured if desired. The processor runs user code and uses the FPGA to handle hardware-related library calls. Both of these boards contain ports for connecting with complementary metal–oxide–semiconductor (CMOS) cameras.

The XBC uses a Nintendo GameBoy Advance as part of its architecture to provide processor, display, and user buttons. This provides the user with a color graphics screen on the robot that can be used for displaying text and images and for setting parameters. This is especially useful for tuning vision parameters, as the camera output (processed or raw) can be redirected to the screen.

55.4.2 Edutainment Programming Environments

Many of the lower-priced controllers (e.g., BASIC Stamp) have limited options for programming environments, while the higher-end controllers such as the XBC can accept cross-compiled programs in most languages.

The LEGO processors are shipped with a graphical programming environment. While sometimes limiting in what can be programmed, the environment is highly intuitive. A more capable (and not quite as intuitive) graphical programming environment based on LabVIEW has been created for the LEGO processors and has been used in a number of schools [55.60].

For educational environments where the robotics experience is serving as an introduction to programming, there are a few good choices. There are Java, C, and

other language implementations specific to one or more of the processors above.

One the most popular (because it runs on many of the processors and under all major operating systems and is free) is Interactive C or IC [55.61]. IC provides a fully integrated development environment (IDE) including an editor, compiler/interpreter and controller simulator. IC implements a subset of American National Standards Institute (ANSI) C, the primary missing functionality being in pointer arithmetic. Other differences between IC and standard C are there to protect the novice user (e.g., IC performs array bounds checking during runtime).

IC also has an interpreted interaction window which allows the user to try out C expressions or function calls, query global variables, etc. either to the actual hardware or to the hardware simulator. The IC environment also provides utilities such as uploading global array data from the robot hardware to the PC, providing memory maps of the program on the controller, colorization, block-level indentation, and interactive documentation from the editor. The IC runtime environment provides library functionality for multithreading.

IC is used regularly in a number of the robotics tournaments described in Sect. 55.1. It is also being widely used in introductory programming courses [55.62] robotics courses [55.63] and for more advanced courses such as in the teaching of neural nets and control [55.64].

IC uses a virtual machine model that runs on the processor, allowing it to be ported to new robot controllers with relative ease.

55.5 Robots and Informal Learning Venues (Museums)

Robots are objects of great attraction and can have a large number of modalities for establishing a strong interaction with humans. It is therefore not surprising that one of the first applications of mobile robots have been robot guides in research labs and later in museums and exhibitions. SHAKEY (Fig. 55.10a), developed at SRI in the early 1980s is often considered the first mobile robot [55.65]. It was able to move wooden blocks according to verbal commands given via a keyboard. Probably the first robot exposed to untrained visitors was POLLY, developed to give simple lab tours at MIT's artificial intelligence (AI) laboratory in the early 1990s [55.66]. The only communication input was vision; visitors waved their feet if they wanted a lab tour. Whereas these early solutions were landmarks in

their time, more recent tour robots offer a much richer interaction and are able to navigate in the somewhat more complex and dynamic environment of museums and exhibitions. One of the first examples of these next-generation tour-guide robot prototypes is Jijo-2, demonstrated in the late 1990s [55.67] (Fig. 55.10c). However, it was operated under laboratory conditions and only by people that were more or less familiar with the robot.

55.5.1 Tour-Guide Robot Examples

The first real tour-guide robot, called Rhino, was deployed for six days in the Deutsches Museum in Bonn during the summer of 1997. It was freely navigating

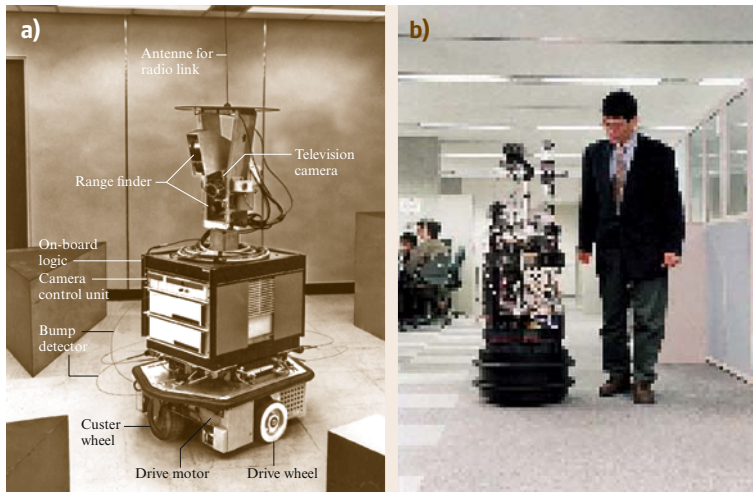


Fig. 55.10a,b The SHAKEY Robot developed in the 60-ties at SRI (a) and the Jijo-2 Robot giving a tour at the National Institute of Advanced Industrial Science and Technology (AIST) (b)

and equipped with speech synthesis (output) and buttons for user input. The robot's main tasks were to get people involved and to support them in navigating through the environment. The experience gained in this short-term installation showed a strong need for enhanced human-robot interaction [55.68] (Fig. 55.11a).

A first step towards a more appealing interaction was realized with Minerva, which was operated for two weeks in late summer 1998 in the Smithsonian's National Museum of History, Washington DC, USA [55.69] (Fig. 55.11b). One version of Minerva was equipped with a caricature face (eyes, eyebrows, mouth), enabling it to appear to express its feelings and entertain the visitors. In addition to on-site interaction, Minerva was accessible for tours via the

Internet from remote sites. According to a study, people perceived the robot's intelligence similar to that of a dog. This basic experience was further expanded in a European project towards two installations of tour-guide robots enabling telepresence in museums and fairs [55.70]. The robot Albert, developed in this project, was equipped with a commercially available speech recognition system that enabled on-site visitors to give spoken commands.

Hermes is a robot from the Universität der Bundeswehr München that was demonstrated at various fairs and exhibitions in 2001 and 2002 [55.71] (Fig. 55.12a). Equipped with two arms, Hermes is able to conduct simple fetch and carry tasks. However, his mobility was typically limited to a very small and well-structured area.

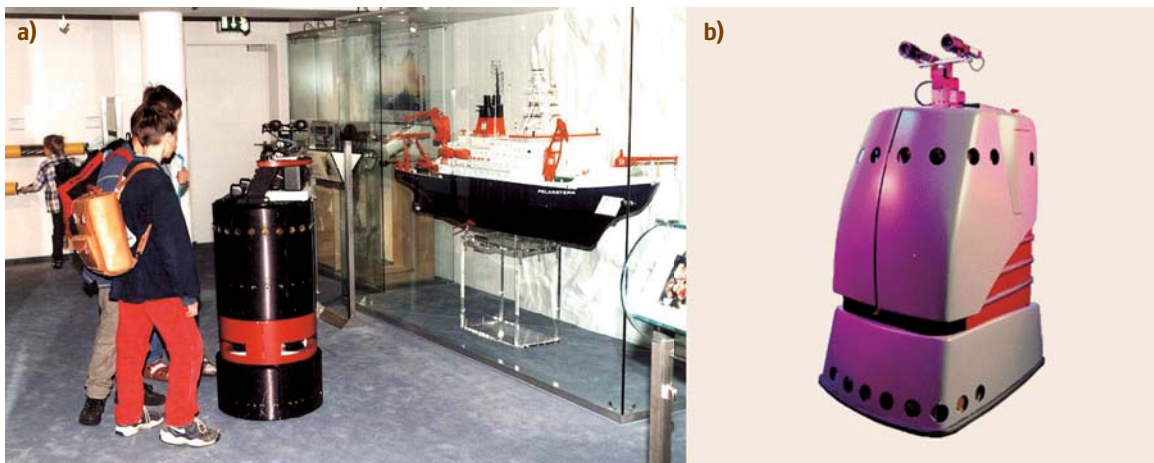


Fig. 55.11a,b Rhino (a), (University of Bonn) and Minerva (b), (Carnegie Mellon University, University of Bonn)



Fig. 55.12a,b Chips developed at Carnegie Mellon University (a) and Hermes, a humanoid interaction robot developed at Universität der Bundeswehr München (b)

Using an off-board microphone, Hermes was able to get simple spoken commands based on keywords.

The first robots deployed in a museum for a long period were Chips (Fig. 55.12b) and its successors SWEETLIPS and JOE, developed by Carnegie Mellon University and Mobot Robotics [55.72, 73]. Installed in 1998, they were operated for several years in the Carnegie Museum of National History, the Hall of North America Wildlife, and the Heinz History Center, respectively. All robots used specially deployed color landmarks for localization, and a touch screen for en-

hanced audiovisual interface. The robots primary goal was to attract visitors to less frequented areas and to enhance the learning rate during the visit. They were conceived as educators offering background information about the exhibition through their audiovisual touch screen. A study done along with the installation of Chips showed that the learning success rate increased by more than 50% when visitors (children and adults) were guided by the robot through the exhibition [55.72].

The Inciting, the Instructive, and the Twiddling robots were developed by the Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) in Stuttgart (Fig. 55.13) [55.74]. These very appealing robots use old-fashioned technology elements such as a mechanical scale, as design elements, and have been running in the Communication Museum of Berlin since 2000. They attract and greet visitors (The Inciting), offer introductions to the museum (The Instructive), and play with a big ball (The Twiddling). They have no input device to give commands, but are very expressive and playful through their appearance and body motions.

The largest installation of tour-guide robots, in terms of number of robots and accumulated operation hours, was realized with a fleet of 11 RoboX robots at the Swiss National Exhibition Expo 02 in 2002 [55.75, 76] (Fig. 55.14). The robots presented the robotics exhibition and themselves through synthesized speech in four languages. A special programming interface enabled easy implementation of different scenarios using speech and sounds, facial expressions (two independent pan-tilt eyes, eyebrows), display of icons in one eye, body motion, and navigation as output modality, and the buttons, face tracking, people tracking and an analysis of the global visitor distribution in the exhibition as input modality. Two robots were equipped with speech recognition enabling simple spoken commands like yes/no in four languages [55.77]. After finishing their job at Expo 02, the RoboX robots found a second life as research platforms and were installed in other exhibitions and as TV moderators by BlueBotics.

Rackham is a tour-guide robot deployed in 2004 at the Mission Biospace Exhibition in Toulouse, France [55.78, 79] (Fig. 55.15a). This robot, developed by the Laboratoire d'Architecture et d'Analyse des Systèmes (LAAS) in Toulouse, has an enhanced audiovisual output system and obtains user commands via a touch screen. The animated helmet together with the touch screen are used as a focal point of attraction to this very futuristic robot. Once a visitor is detected by the pan-tilt camera in the helmet, the robot explains its utilization



Fig. 55.13 The Inciting, the Instructive, and the Twiddling robot amusing and informing peoples at the Communication Museum, Berlin. These very attractive robots have been developed at Fraunhofer Institute for Manufacturing Engineering and Automation (IPA)

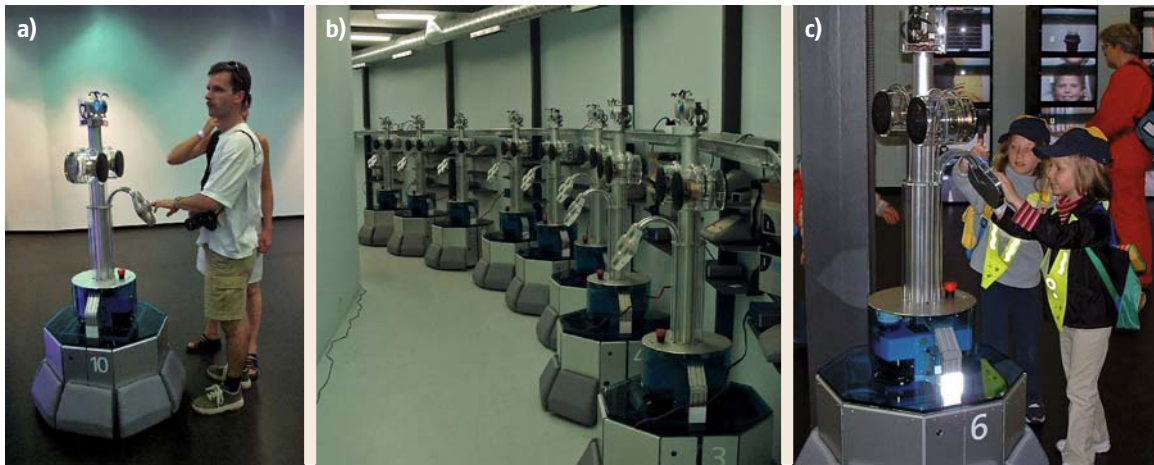


Fig. 55.14a–c The family of 11 RoboX tour guide developed by the EPFL gave tours during five months to over 650,000 visitors at the Swiss National exhibition Expo 02

and offers its services. This installation is also used as an experimental environment for further research.

More recent tour-guide robot platform developments are focusing on enhanced human–robot interaction and more humanoid appearance. One illustration is Brion, which learns the environmental setting from humans through an enhanced dialog system [55.80]. Voice and gesture recognition are the main inputs for driving the dialogue with the human that is based on a finite state machine (FSM). The humanoid exhibition robot Repliee Q1 has been developed at Osaka University. It consists of an artificial human upper body with a very realistic appearance (Fig. 55.15b). Repliee Q1 communicates through speech, enhanced facial expressions and body language. It has a tactile skin over the entire body, able to sense being touched and react to it. During the 2005 World Expo in Japan, Repliee Q1 was offering its services at an information desk. The next generation of Repliee is intended to have actuated legs for full motion abilities.

More sophisticated robots will be deployed in informal learning venues as tour guides and teachers soon. They have the potential to offer a more appealing and lasting learning experience. However, there is still important research and technological advancement required. Until then, robots can only partially compete with a human guide.

55.5.2 Lessons Learned

Through the various installations presented above, new insights on the design of tour-guide robots and the user's



Fig. 55.15a,b Rackham is a tour-guide robot deployed at the Mission Biospace Exhibition in Toulouse, France (a). The Repliee Q1 humanoid robot (b) was at the information counter at the 2005 World Expo in Japan. It was developed by Osaka University

behaviors were gained. Various details can be found in [55.69, 72] and [55.76]. At informal learning venues it is nearly impossible to instruct the visitors on how to operate the robots. Thus the robot's interactivity has to be plug-and-play. This is only possible if design and functionality are harmonically integrated and the robot's intention and internal states are transparent and understandable for the visitor. Design creates expectations, e.g., a humanoid appearance makes visitors believe that the robot has similar competence as a human. A robot that speaks is also expected to understand spoken dialog. In general, the public has the tendency to overestimate the robot's competence because most people only know

robots from science-fiction movies or futuristic research documentaries.

Visitors are curious, but also impatient and rude. Therefore tour-guide robots have to allow visitors to discover new elements in a fast rhythm, have to be very dynamic and fast responding, and should be robust against harsh treatment. If possible, visitors will sit and hang on the robot or hit the bumpers. On the other hand, visitors are respectful with elements that look fragile, e.g., they will not touch the eyes of the robot. The most appealing interactions are often elements such as

obstacle avoidance or even security buttons because the feedback is immediate and easy to understand. It is therefore of crucial importance that the actions of the robot are immediately responsive to users' needs and wishes. A delay of only one second to a pushed button is often too much and will often incite inappropriate reactions from the visitors, such as hammering on the button.

A successful installation of robots in informal learning venues must be carefully designed according to the functionalities and services they are intended to offer. Form has to follow function.

55.6 Educational Evaluation of Robot Programs

An important aspect of assessment, beyond technical evaluation, is educational evaluation: how does human–robot interaction change models of learning and identities with respect to technology; and how is this done in educationally appropriate ways? We find several techniques to be applicable during each of the following phases of an educational project:

- design-time assessment: the creation phase of an educational robot system
- formative evaluation: during execution of an educational robot program
- summative evaluation: following completion of an educational program

This section will identify relevant tools for design, formative, and summative evaluation that we borrow from the human–computer interaction community and the cognitive-educational analysis community. These tools are well suited to helping researchers understand and quantify how educational robots can impact learning in the classroom and, beyond the classroom, in informal learning venues such as museums and after-school workshops.

55.6.1 Design-Time Assessment

The field of human–computer interaction (HCI) offers formal and heuristic techniques for addressing interfaces between humans and technology [55.81, 82]. Current HCI practice often focuses on user-centered design, which suggests that understanding the user and the entire task can help us be better designers. One specific HCI technique, contextual inquiry and design, is especially useful for both informal and formal learning venues where robots will interact directly with students

as users. Contextual inquiry guides the collection of background data concerning interactions taking place before inclusion of the new, to-be-designed robotic technology, and is often achieved through the collection and analysis of direct interviews and observations. The process includes identifying and polling various stakeholders in the educational environment, from the classroom as a whole to individual educators and mentors.

A second HCI activity, modeling, guides the creation of pictorial flow diagrams that identify the most important features of the environment in which the robotic educational technology is to be used. *Workflow* describes the communication necessary for the classroom activities. *Sequence* identifies actual sequences of action required for various activities in the classroom (or museum). *Artifact* captures the relationship of existing tools required for activities to be performed successfully. Finally the *physical model* identifies how space is used, which can be notable in the case of the design and potential inclusion of mobile robots in spaces that were generally not designed for them in the first place, such as conventional laboratories and, even worse, regular lecture halls.

Why do all the above? Because it is a principled way of understanding existing environments. This in turn facilitates design-time judgments in regards to the best manner in which to add a new technology, such as interactive robots, while maximizing the chances that the technology is a real success. However, it is the continual cycle of evaluations during the implementation phase that is critical to a positive outcome. So in summary design-time data collection and assessment helps set the stage and control expectations. Formative evaluations and feedback into the continuing design process,

as described below, help keep the engineering on a path to success.

55.6.2 Formative and Summative Evaluation

Traditional school-based assessments of learning, for instance course surveys or exam grades, are useful as coarse-resolution data collection devices. However, these do not provide sufficient detail to build models of how learning is proceeding with or despite of technology changes, nor how further technology changes may improve learning. Furthermore, especially in informal settings such as museums, school-based assessments are even more inappropriate. In such settings significant learning results from communication. As groups of visitors or teams of students use and talk about a museum exhibit or project challenge, they construct a shared understanding of the content and context of the challenge. There is an outstanding body of work in the education community that focuses on this form of conversation and learning. The following references are excellent starting points [55.83–87]. Applying this education learning strategy to educational robotics yields wonderfully rich analyses in terms of how the robots change conversations in the classroom [55.88].

In education learning, tools include broad evaluation tools for qualitatively and quantitatively constructing a model of learning across the classroom and deep, focused tools for studying the specific communication-oriented mechanisms of learning that may inform patterns of change and exploration at the student and team level. In terms of broad tools, written student surveys are one useful data collection tool. A combination of linear, quantitative queries (called Likert scales) coupled with open-ended essay questions such as *What is the most important concept you learned this week?* invite both statistical analysis of raw results and thematic analysis of self-reported challenges and successes by the students themselves. Recorded interviews with students, individually or in teams, provides even richer information due to the ability to ask complex *What have you learned?* questions appropriate for open-ended analysis, again in terms of thematic content. A deep, focused tool that is expensive in terms of analyst resources but generates very useful information is the ethnography, whereby a trained observer spends, for instance, one week tracking all the activities of two teams in the classroom, including conversations within the team, among team members and other pupils, and between the team and the course instructors and teaching as-

sistants [55.89–91]. Ethnography is intended to detail the learning and problem-solving process that occurs in class on a minute-by-minute basis.

Taken as a whole, the above collected data can then be analyzed by thematic content, for themes such as those listed in Sect. 55.1. Statistical significance and correlation tests can be applied to the quantitatively derived results of coding all such information to yield information on how the frequency of learning themes appearing in conversation increases or decreases, and how specific themes become more refined or specific in students' speech over the duration of the course.

However, assessment of the educational value of robotics programs such as those described in Table 55.1 is often more difficult than evaluating programs in more traditional disciplines. There are widely accepted textbooks and curricula for many disciplines, but not yet for robotics. Additionally, the different educational robotics programs described earlier take a variety of approaches towards education.

Many programs (e.g., FIRST LEGO League and RoboCup Jr) are large but are implemented at the local level, so the experience for students can vary tremendously from one locale to another. The more standardized programs (e.g., Botball and FIRST) have different educational models. FIRST uses an inspirational model where students work with professional engineers. Botball follows a more traditional high-school model of education where the teachers are trained in the techniques and principles, so that they may guide the students at a high level, but all of the work (both design and implementation) is done by the students.

Because robotics can serve as an unfamiliar tool for many students in such courses and programs, evaluation of how this unfamiliar, potentially rich interactive tool impacts student learning and perceived interest levels in science and technology is a common focus of educational robotics assessment. This trend is exacerbated because these programs, which involve tens or even hundreds of thousands of students, are implemented by nonprofit companies and volunteers that lack specific funds and training for formal assessment. For these reasons, formal assessment of the programs tends to be based on student and/or teacher surveys rather than rigorous testing, and the assessment usually focuses on specific issues (e.g., gender issues in technology education) where assessment-specific funding may be available.

As educational robotics becomes more mainstream, we can expect that evaluation will shift from general enthusiasm about science, technology, engineering, and

mathematics (STEM) learning to specific learning advantages derived from such robot-inclusive coursework for follow-on coursework in diverse disciplines such as biology, physics, and computer science.

55.7 Conclusions and Further Reading

This chapter has given an overview of many, but by no means all, of the educational robot hardware and programs currently available. Websites such as robots.net and robotevents.com attempt to maintain lists of the rapidly evolving field of educational robot tournaments. Keeping track of the latest hardware and software tools are more problematic. Some additional sources for these topics, and others covered in this

chapter include: *Robots for Kids, Exploring new technologies for learning*, edited by Druin and Hendler; *Robotic Explorations: A Hands-On Introduction to Engineering* [55.92], by Martin; *Where the Action Is, Foundations of embodied interaction*, by Dourish; *Piaget's theory of cognitive development* [55.93], by Wadsworth; and *Computer as Theater* [55.94], by Laurel.

References

55.1 KIPR: Botball robotics education. <http://www.botball.org> (KISS, Norman 2005)

55.2 R. Manseur: Hardware competitions in engineering education, *Front. Educ. Conf.* **2**, F3C/5–F3C/8 (2000)

55.3 Robo Cup Junior: RoboCupJunior official site. <http://www.robocupjunior.org/>, (Acroname Robotics, Boulder 2006)

55.4 C. Stein: Botball: Autonomous students engineering autonomous robots, *Comput. Educ. J.* **13**(2), 72–80 (2003)

55.5 FIRST: <http://www.usfirst.org/> (FIRST, Manchester 2006)

55.6 A. Druin, J. Hendler (Eds.): *Robots for Kids: Exploring New Technologies for Learning* (Morgan Kaufmann, San Francisco 2000)

55.7 R. Siegwart: Grasping the interdisciplinarity of mechatronics, *IEEE Robot. Autom. Mag.* **8**(2), 27–34 (2001)

55.8 R.D. Beer, H.J. Chiel, R.F. Drushel: Using Autonomous Robotics to teach science and engineering, *Commun. ACM* **42**(6), 85–92 (1999)

55.9 A. Billard: Robota, clever toy and educational tool, *Robot. Autom. Syst.* **42**, 259–269 (2003)

55.10 P. Coppin: Eventscope: a telescience interface for internet-based education, *IEEE Proc. Robot. Autom.* (2002)

55.11 B. Fagin: Ada/mindstorms 3.0: A computational environment for introductory robotics and programming, *IEEE Robot. Autom. Mag.* **10**(2), 19–24 (2003)

55.12 R.S. Hobson: The changing face of classroom instructional methods: service learning and design in a robotics course, *Frontiers in Education Conference*, Vol. 2 (Kansas City 2000) pp. F3C/20–F3C/25

55.13 D. Kumar, L. Meeden: A robot laboratory for teaching artificial intelligence, *SIGCSE '98: Proc.* 29th SIGCSE Tech. Symp. Comp. Sci. Edu. (ACM, New York 1998) pp. 341–344

55.14 J. Schumacher, D. Welch, D. Raymod: Teaching introductory programming, problem solving and information technology with robots at west point, *Proc. 31st Frontiers in Education Conference*, Vol. 2 (Reno 2001)

55.15 E. Wang: Teaching freshmen design, creativity and programming with legos and labview, *Front. Educ. Conf.*, F3G–11–15 (2001)

55.16 U. Wolz: Teaching design and project management with lego rcx robots, *SIGCSE '01: Proc. 32nd SIGCSE Tech. Symp. Comp. Sci. Edu.* (ACM, New York 2001) pp. 95–99

55.17 J.K. Archibald, R.W. Beard: Goal! robot soccer for undergraduate students, *IEEE Robot. Autom. Mag.* **11**(1), 70–75 (2004)

55.18 A. Gage, R.R. Murphy: Principles and experiences in using legos to teach behavioural robotics, *Proc. 33rd Frontiers in Education Conference*, Vol. 2 (Sarasota 2003)

55.19 E. Kolberg, N. Orlev: Robotics learning as a tool for integrating science technology curriculum in k-12 schools, *Front. Educ. Conf.* **1**, T2E–12–13 (2001)

55.20 B.A. Maxwell, L. Meeden: Integrating robotics research with undergraduate education, *IEEE Intell. Syst.* **15**(6), 22–27 (2000)

55.21 A. Nagchaudhuri, G. Singh, M. Kaur, S. George: Lego robotics products boost student creativity in pre-college programs at umes, *Front. Educ. Conf.* **3**, S4D–1–S4D–6 (2002)

55.22 C. Stein, K. Nickerson: Botball robotics and gender differences in middle school teams, *Proc. ASEE Annual Conference* (Salt Lake City 2004)

55.23 J.B. Weinberg, J.C. Pettibone, S.L. Thomas, M.L. Stephen, C. Stein: The impact of robot projects

- on girls' attitudes toward science and engineering, Proc. RSS Robotics in Education Workshop (Atlanta 2007)
- 55.24 T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of socially interactive robots (2002)
- 55.25 F. Martin, B. Mikhak, M. Resnick, B. Silverman, R. Berg: *To Mindstorms and Beyond: Evolution of a Construction Kit for Magical Machines* (Morgan Kaufmann, San Francisco 2000) pp. 10–32
- 55.26 M. Cooper, D. Keating, W. Harwin, K. Dautenhahn: *Robots in the Classroom – Tools for Accessible Education* (IOS, Amsterdam 1999)
- 55.27 6.270 Organizers: The history of 6.270 – MIT's autonomous robot design competition. <http://web.mit.edu/6.270/www/about/history.html> (MIT, Cambridge 2005)
- 55.28 BEST Robotics: Middle and high school robotics competition. <http://www.bestinc.org/MVC/> (BEST Robotics Inc., Auburn 2006)
- 55.29 D. Ahlgren: Trinity college fire fighting home robot contest. <http://www.trincoll.edu/events/robot/> (Trinity College, Hartford 2006)
- 55.30 Fuji Soft ABC: FSI—all japan robot-sumo tournament, <http://www.fsi.co.jp/sumo-e/> (Fuji Soft Inc., Tokyo 2006)
- 55.31 IMRMC Organizers: International micro robot maze contest. <http://imd.eng.kagawa-u.ac.jp/maze/> (Nagoya Univ., Nagoya 2006)
- 55.32 Robocup official site: <http://www.roboocup.org/> (The RoboCup Federation, Osaka 2007)
- 55.33 D. Nardi, M. Riedmiller, C. Sammut, J. Santos-Victor (Eds.): RoboCup 2004: Robot Soccer World Cup VIII. In: *Lecture Notes in Computer Science*, Vol. 3276 (Springer, Berlin/Heidelberg 2005)
- 55.34 T.R. Balch, H.A. Yanco: Ten years of the aaai mobile robot competition and exhibition, *Artif. Intell. Mag.* **23**(1), 13–22 (2002)
- 55.35 C. Pomalaza-Raez, B.H. Groff: Retention 101: Where robots go...students follow, *J. Eng. Educ.* **92**(1) (2003)
- 55.36 D.P. Miller, C. Winton, J.B. Weinberg: *Beyond Botball: A Software Oriented Robotics Challenge for Undergraduate Education. Techn. Rep. SS-07-09* (AAAI, Menlo Park 2007)
- 55.37 R.R. Murphy: Using robot competitions to promote intellectual development, *Artif. Intell. Mag.* **21**(1), 77–90 (2000)
- 55.38 D.P. Miller, C. Stein: *So that's what Pi is for! And other Educational Epiphanies from Hands-on Robotics* (Morgan Kaufmann, San Francisco 2000) pp. 219–243
- 55.39 I. Vernor, D. Ahlgren, D.P. Miller: Robotics olympiads: A new means to integrate theory and practice in robotics, Proc. ASEE 2006 (Chicago 2006)
- 55.40 NASA Robotics Alliance Project: Education matrix. <http://robotics.nasa.gov/edu/matrix.php> (NASA, Washington 2006)
- 55.41 K-Team: <http://www.k-team.com> K-Team Corp, Yverdon les bains, K-Team Corp Switzerland (2007)
- 55.42 PPRK: Palm Pilot Robot Kit, <http://www.cs.cmu.edu/pprk/> Carnegie Mellon University, Pittsburgh (2007)
- 55.43 E-puck: <http://www.e-puck.org> (EPFL, Lausanne 2007)
- 55.44 Garcia Robot: <http://www.acroname.com> (Acroname Robotics, Boulder 2007)
- 55.45 ER1: <http://www.evolution.com> (Evolution Robotics, Pasadena 2007)
- 55.46 KHR-1: <http://www.kondo-robot.com> (Kondo Kagaku, 2007)
- 55.47 Robsapien: <http://www.robosapien.com> (WowWee, Quebec 2007)
- 55.48 Roomba: <http://www.irobot.com> (iRobot Corp., Burlington 2007)
- 55.49 iRobot Corp: *iRobot Roomba Serial Command Interface (SCI) Specification* (iRobot, Burlington 2005)
- 55.50 T.E. Kurt: *Hacking Roomba* (Wiley, New York 2006)
- 55.51 D.P. Miller: Quick and easy way to add vision to your irobot create or roomba. <http://i-borg.engr.ou.edu/dmiller/create/>, Oklahoma University (2007)
- 55.52 Gumstix: <http://www.gumstix.com> (Gumstix Inc., Portola Valley 2007)
- 55.53 M.J. Matarić, N. Koenig, D. Feil-Seifer: *Materials for Enabling Hands-on Robotics and STEM Education, Tech. Rep. SS-07-09* (AAAI, Menlo Park 2007)
- 55.54 J.L. Jones, A.M. Flynn: *Mobile Robots: Inspiration to Implementation* (Peters, Wellesley 1993)
- 55.55 BASIC stamps: <http://www.parallax.com/htmlpages=products=basicstamps=basicstamps.asp> (Parallax Inc., Rooklin 2007)
- 55.56 Savage Innovations: OOPic home page. <http://www.oopic.com/> (Savage Innovations, Huntsville 2007)
- 55.57 S. Richards: Brain stem gp 1.0 module. <http://www.acroname.com/robotics/parts/S1-GP-BRD.html>
- 55.58 R. LeGrand, K. Machulis, D.P. Miller, R. Sargent, A. Wright: The XBC: A modern low-cost mobile robot controller, IEEE Proc. IROS 2005 (Edmonton 2005)
- 55.59 F. Martin: The black fin handyboard. <http://www.cs.uml.edu/blackfin/> (Analog Devices, Norwood 2007)
- 55.60 C. Rogers, M. Portsmore: Bringing engineering to elementary school, *J. STEM Educ.* **5**(3, 4), 17–28 (2004)
- 55.61 Interactive C: <http://www.kipr.org/ic> (KISS Inst. for Practical Robots, Norman 2007)
- 55.62 C.E. van Lent: Using robot platforms to enhance concept learning in introductory cs courses, AAAI Spring Symp. Robotics in Education (Stanford 2004)
- 55.63 D.P. Miller, C. Winton: Botball kit for teaching engineering computing, Proc. ASEE National Conference (ASEE, Salt Lake City 2004)
- 55.64 S.P. Imberman: Teaching neural networks using lego handy board robots in an artificial intelligence

- course, SIGCSE '03: Proc. 34th SIGCSE Tech. Symp. Comp. Sci. Edu. (ACM, New York 2003) pp. 312–316
- 55.65 N. Nilson: *Shakey the Robot, Techn. Rep. 232* (SRI Int., Menlo Park 1984)
- 55.66 I. Horswill: Visual support for navigation in the polly system, Proc. AAAI Fall Symposium on Applications of Artificial Intelligence to Real-World Autonomous Mobile Robots (Cambridge 1992) pp. 62–67
- 55.67 T. Matsui, H. Asoh, J. Fry, Y. Motomura, F. Asano, T. Kurita, I. Hara, N. Otsu: Integrated natural spoken dialogue system for jijo-2 mobile robot for office services, AAAI/IAAI (1999) pp. 621–627
- 55.68 W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, S. Thrun: Experiences with an interactive museum tour-guide robot, *Artif. Intell.* **114**(1–2), 3–55 (1999)
- 55.69 S. Thrun, M. Bennewitz, W. Burgard, A. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, D. Schulz: Minerva: a second-generation museum tour-guide robot, Proc. Int. Conf. Robot. Autom. (ICRA) **3**, 1999–2005 (1999)
- 55.70 W. Burgard, P. Thrahanis, D. Hahnel, M. Moors, D. Schulz, H. Baltzakis, A.A. Tourbot: webfair: Web-operated mobile robots for tele-presence in populated exhibitions, Workshop: Robotic in Exhibition, IROS 2002 (Lausanne 2002)
- 55.71 R. Bischoff: Natural communication and interaction with humanoid robots, Proc. the 2nd Int. Symposium on Humanoid Robots (Tokyo 1999) pp. 121–128
- 55.72 I.R. Nourbakhsh: Nourbakhsh: The mobot museum robot installations: A five year experiment, Workshop: Robotic in Exhibition IROS 2002 (Lausanne 2002)
- 55.73 T. Willeke, C. Kunz, I. Nourbakhsh: The history of the Mobot museum robot series: An evolutionary study, Proc. of the Florida Artificial Intelligence Research Society (FLAIRS) (Key West 2001) pp. 514–518
- 55.74 B. Graf, O. Barth: Entertainment robotics: Examples, key technologies and perspectives, Workshop: Robotic in Exhibition, IROS 2002 (Lausanne 2002)
- 55.75 R. Siegwart, K. Arras, S. Bouabdallah, D. Burnier, G. Froidevaux, X. Greppin, B. Jensen, A. Lorotte, L. Mayor, M. Meisser, R. Piguët, G. Ramel, G. Térien, N. Tomatis: Robox at expo.02: A large scale installation of personal robots, *Robot. Auton. Syst.* **42**(3–4), 203–222 (2003)
- 55.76 B. Jensen, N. Tomatis, L. Mayor, A. Drygajlo, R. Siegwart: Robots meet humans – interaction in public spaces, *IEEE Trans. Ind. Electron.* **52**(6), 1530–1546 (2006)
- 55.77 A. Drygajlo, P. Prodanov, G. Ramel, M. Meisser, R. Siegwart: On developing voice enabled interface for interactive tour-guide robots, *Adv. Robot.* **17**(7), 599–616 (2003)
- 55.78 R. Alami: Towards a personal robot. In Workshop: Robotic in Exhibition, IROS 2002, Lausanne (2002)
- 55.79 A. Clodic, S. Fleury, R. Alami, M. Herrb, R. Chatila: Supervision and interaction: Analysis for an autonomous tour-guide robot deployment, Proc. of IEEE International Conference on Advanced Robotics, ICAR (Seattle 2004)
- 55.80 A. Haasch, A. Hohenner, S. Huwel, M. Kleinehagenbrock, S. Lange, I. Tóptsis, G. Fink, J. Fritsch, B. Wrede, G. Sagerer: Biron – the bielefeld robot companion, Proc. of International Workshop on Advances in Service Robotics (Stuttgart 2004) pp. 27–32
- 55.81 S. All, I. Nourbakhsh: Insect telepresence: Using robotic tele-embodiment to bring insects face-to-face with humans, *Auton. Robot.* **10**, 149–161 (2001)
- 55.82 B. Laurel (Ed.): *The Art of Human Computer Interface Design* (Addison-Wesley, Reading 1994)
- 55.83 S. Allen: Looking for learning in visitor talk: A methodological exploration. In: *Learning Conversations in Museums*, ed. by G. Leinhardt, K. Crowley, K. Knutson (Lawrence Erlbaum, Mahwah 2002) pp. 259–303
- 55.84 K. Crowley, M. Callanan: Describing and supporting collaborative scientific thinking in parent-child interactions, *J. Museum Educ.* **23**(1), 12–17 (1998)
- 55.85 K. Crowley, M.A. Callanan, J.L. Jipson, J. Galco, K. Topping, J. Shrager: Shared scientific thinking in everyday parent-child activity, *Sci. Educ.* **85**(6), 712–732 (2001)
- 55.86 G. Leinhardt, K. Crowley: Objects of learning, objects of talk: Changing minds in museums. In: *Perspectives on Object-Centered Learning in Museums*, ed. by S.G. Paris (Lawrence Erlbaum, Mahwah 2002)
- 55.87 G. Leinhardt, K. Crowley, K. Knutson (Eds.): *Learning Conversations in Museums* (Lawrence Erlbaum, Mahwah 2002)
- 55.88 I. Nourbakhsh, E. Hamner, D. Bernstein, K. Crowley, E. Ayoob, M. Lotter, S. Shelly, T. Hsiu, E. Porter, B. Dunlavy, D. Clancy: The personal exploration rover: Educational assessment of a robotic exhibit for informal learning venues, *Int. J. Eng. Educ.* (2006) Special Issue on Robotics Education
- 55.89 K. Crowley, M.A. Callanan, H.R. Tenenbaum, E. Allen: Parents explain more often to boys than to girls during shared scientific thinking, *Psychol. Sci.* **12**(3), 258–261 (2001)
- 55.90 H.H. Lund, P. Marti: Physical and conceptual constructions in advanced learning environments, *Interact. Stud.* **5**(2), 271–301 (2004)
- 55.91 I.R. Nourbakhsh, K. Crowley, A. Bhawe, E. Hamner, T. Hsiu, A. Perez-Bergquist, S. Richards, K. Wilkinson: The robotic autonomy mobile robotics course: iRobot design, curriculum design and educational assessment, *Auton. Rob.* **18**(1), 103–127 (2005)
- 55.92 F.G. Martin: *Robotic explorations: A hands-on introduction to engineering* (Prentice Hall, Upper Saddle River 2001)

- 55.93 B.J. Wadsworth: *Piaget's theory of cognitive development: An introduction for students of psychology and education* (McKay, New York 1971)
- 55.94 B. Laurel: *Computers as theatre* (Addison-Wesley, Reading 1993)

Human-Part G

Part G Human-Centered and Life-Like Robotics

Ed. by Daniela Rus

56 Humanoids

Charles C. Kemp, Atlanta, USA
 Paul Fitzpatrick, Genova, Italy
 Hirohisa Hirukawa, Tsukuba, Japan
 Kazuhito Yokoi, Tsukuba, Japan
 Kensuke Harada, Tsukuba, Japan
 Yoshio Matsumoto, Suita, Osaka, Japan

57 Safety for Physical Human–Robot Interaction

Antonio Bicchi, Pisa, Italy
 Michael A. Peshkin, Evanston, USA
 J. Edward Colgate, Evanston, USA

58 Social Robots that Interact with People

Cynthia Breazeal, Cambridge, USA
 Atsuo Takanishi, Tokyo, Japan
 Tetsunori Kobayashi, Tokyo, Japan

59 Robot Programming by Demonstration

Aude Billard, Lausanne, Switzerland
 Sylvain Calinon, Lausanne, Switzerland
 Rüdiger Dillmann, Karlsruhe, Germany
 Stefan Schaal, Los Angeles, USA

60 Biologically Inspired Robots

Jean-Arcady Meyer, Paris, France
 Agnès Guillot, Paris, France

61 Evolutionary Robotics

Dario Floreano, Lausanne, Switzerland
 Phil Husbands, Falmer, UK
 Stefano Nolfi, Roma, Italy

62 Neurorobotics: From Vision to Action

Michael A. Arbib, Los Angeles, USA
 Giorgio Metta, Genova, Italy
 Patrick van der Smagt, Wessling, Germany

63 Perceptual Robotics

Heinrich H. Bülthoff, Tübingen, Germany
 Christian Wallraven, Tübingen, Germany
 Martin A. Giese, Bangor, UK

64 Roboethics: Social and Ethical Implications of Robotics

Gianmarco Veruggio, Genova, Italy
 Fiorella Operto, Sampierdarena, Genova, Italy

Part G Human-Centered and Life-Like Robotics, covers some of the most recent advances concerned with human–robot interaction, ranging from designing biologically inspired robots, to programming and safety issues for human–robot interaction, and ethical issue brought forth by robotics. Our field’s future vision for technology is the leap from personal computers to personal robots, in a world where robots exist pervasively and work side by side with humans. Over the past 50 years we have made great strides in robotics, as the previous parts of this Handbook show. However, there are still new capabilities that need to be developed and existing capabilities that need to be improved to create a world in which robots and humans work together. Robot bodies should be easily integrated into our living environments. Robots should be safe to be around. Robots should take commands from human users easily. Robots should be functionally capable. Robots should engage humans to help mitigate error states and task uncertainties. Meeting these challenges will bring robots closer to our vision of pervasive robotics.

The topics in Part G are essential for creating robots that operate in human-centered environments. The chapters cover organically human-centered and life-like robots and include hardware design, control (of locomotion, manipulation, and interaction), perception, user interfaces, and social and ethical implications for robotics. As such, Part G builds on all of the Handbook’s previous parts. The connection with the chapters in Robot Foundations (Part A) and Robot Structures (Part B), particularly Mechanisms and Actuation (Chap. 3), Sensing and Estimation (Chap. 4), Motion Planning (Chap. 5), Robot Systems Architectures and Programming (Chap. 8), AI Reasoning Methods for Robotics (Chap. 9), Robot Hands (Chap. 15), Legged Robots (Chap. 16), and Wheeled Robots (Chap. 17) are essential. There is also an important dependency on Sensing and Perception (Part C), Manipulation and Interfaces (Part D), and Mobile and Distributed Robotics (Part E) as these chapters cover some of the basic algorithms and technologies required by robots operating in human-centered environments.

Today’s approach to computation has progressed naturally from desktop computing, to mobile computing, to pervasive computing, ultimately leading to computation for interaction with the physical world, in other words, robotics. Part G presents a snapshot of the field’s advances for creating machines in our own image that are smart and obedient.

With this overview of Part G, we now provide a brief synopsis of each chapter.

Chapter 56, Humanoids, provides a concise overview of the state of the art in creating human-like machines, starting with a historical and philosophical discussion on the human example and understanding intelligence. These machines are capable of bipedal locomotion and manipulation. In addition, humanoid robots can take advantage of their entire body (e.g., lifting objects while moving the body to compensate for the load). Control issues specific to humanoids include static and dynamic stability during a mobile manipulation action. Humanoid robots interact with humans via perception systems which should be capable of interpreting behavior and mood as well as via speech systems for interpreting human language.

Chapter 57, Safety for Physical Human–Robot Interaction, discusses what happens when humans and robots share the same workspace and come into contact with one another. Safety can be jeopardized by mechanical failures, operator errors, or software problems. Designing and controlling robots with safety guarantees is a very important precondition for physical human–robot interaction (pHRI). The chapter discusses safety standards for pHRI and surveys several approaches to safety.

Chapter 58, Social Robots that Interact with People, discusses the creation of robots designed to interact with people in a social and emotional way for applications in education, entertainment, health care, etc. These robots’ embodiment is life-like and they are capable of multi-modal communication. The chapter uses a case study of the Kismet robot to illustrate the idea of a machine capable of projecting emotion. The chapter surveys some basic approaches to giving robots sociocognitive skills such as shared attention, emotional empathy, and mental perspective tasking.

Chapter 59, Robot Programming by Demonstration, is an important approach to teaching robots what to do by providing either good or bad examples. This is an intuitive approach to robot control and learning of tasks with the potential of opening robot applications to nonexpert users. The chapter starts by discussing what it means for a robot to learn a new skill and surveys several learning approaches for this. Biologically oriented learning approaches such as conceptual models of imitation learning and neural models of imitation learning are also introduced. The chapter concludes with several open issues in robot programming by demonstration.

Chapter 60, Biologically Inspired Robots, starts with a discussion of the difference between bio-inspired and biomimetic robots. Bio-inspired robots seek to re-

produce a natural phenomenon but not necessarily the underlying means. Biomimetic robots reproduce both the natural phenomenon and the means. The chapter starts with a survey of bio-inspired morphologies. A survey of bio-inspired sensors from vision to audio, touch, smell, and taste is then presented. Bio-inspired actuators include locomotion topics such as crawling, legged locomotion, climbing swimming, flying, and manipulation. The chapter also examines bio-inspiration in the context of control and planning. This includes behavior-based architectures, learning robots, evolving robots, and developing robots. Finally, the chapter examines how to make machines self-sustaining from an energy point of view.

Chapter 61, Evolutionary Robotics, discusses a method for creating robots that is inspired by the Darwinian principle of selective reproduction of the fittest captured by evolutionary algorithms. The chapter describes the basic method and presents several case studies. Because the evolutionary method requires intensive computation, it is often run in simulation and the final result is transferred to the robot platform. However, in the recent past there were several successes with running evolutionary algorithms in real time on physical platforms to achieve complex behaviors such as learning to walk.

Chapter 62, Neurorobotics: From Vision to Action, looks at how neuroethology, the study of brain mechanisms for animal behavior, inspires the creation of a central nervous system for a machine. The chapter presents several case studies that sample this bio-inspired design of robot controllers structurally and functionally: the optic flow in bees and robots, visually

guided behaviors in frogs and robots, and navigation in rats and robots. The chapter then examines the role of the cerebellum in motion control and the role of mirror systems. The chapter concludes by observing that the brain has evolved to serve as action-oriented perception, which is a useful guiding principle for building robot controllers.

Chapter 63, Perceptual Robotics, discusses principles derived from high-level cognitive processing in vision in the human brain that have led to results in robotics and computer vision. The chapter is focused on the technical realization of perceptual functions in robots by examining relationships between biological perception and robotics systems. Object recognition is used as an exemplary task and issues such as representation, structural description models, neural representations, as well as recognition and learning algorithms are discussed. Next, example-based movement representations are examined to establish the connection between the functionality of the visual cortex and a class of computer vision algorithms.

Chapter 64, Roboethics: Social and Ethical Implications of Robotics, discusses the social and ethical implications in a society where robots coexist with humans. Starting with a philosophical introspection into the dangers of unlimited use of technology, the chapter argues for the need for a new ethics of robots. Many other factors, including the cultural differences in societies ready to accept robots, define this need. The chapter surveys the code of ethics, privacy, accuracy, intellectual property, and access which could be adopted by robotics. A taxonomy of robots, along with socio-ethical issues for each robot type is then presented.

Humanoids

56. Humanoids

Charles C. Kemp, Paul Fitzpatrick, Hirohisa Hirukawa, Kazuhito Yokoi, Kensuke Harada, Yoshio Matsumoto

Humanoid robots selectively emulate aspects of human form and behavior. Humanoids come in a variety of shapes and sizes, from complete human-size legged robots to isolated robotic heads with human-like sensing and expression. This chapter highlights significant humanoid platforms and achievements, and discusses some of the underlying goals behind this area of robotics. Humanoids tend to require the integration of many of the methods covered in detail within other chapters of this handbook, so this chapter focuses on distinctive aspects of humanoid robotics with liberal cross-referencing.

This chapter examines what motivates researchers to pursue humanoid robotics, and provides a taste of the evolution of this field over time. It summarizes work on legged humanoid locomotion, humanoid manipulation, whole-body activities, and approaches to human-robot communication. It concludes with a brief discussion of factors that may influence the future of humanoid robots.

56.1 Why Humanoids?	1307
56.1.1 The Human Example	1308
56.1.2 The Pleasing Mirror	1308
56.1.3 Understanding Intelligence	1308
56.1.4 Interfacing with the Human World	1308
56.1.5 Interfacing with People	1309
56.1.6 Entertainment, Culture, and Surrogates	1310
56.2 History and Overview	1310
56.2.1 Different Forms	1311
56.2.2 Different Degrees of Freedom	1311
56.2.3 Different Sensors	1311
56.2.4 Other Dimensions of Variation	1312
56.3 Locomotion	1312
56.3.1 Bipedal Locomotion	1312
56.3.2 Falling Down	1313
56.3.3 Sensing for Balance	1314
56.3.4 Localization and Obstacle Detection	1314
56.4 Manipulation	1315
56.4.1 The Arm and Hand	1315
56.4.2 Sensing for Manipulation	1316
56.4.3 Rhythmic Manipulation	1317
56.4.4 Cooperative Manipulation	1317
56.4.5 Learning and Development	1318
56.5 Whole-Body Activities	1318
56.5.1 Coarse Whole-Body Motion	1319
56.5.2 Generating Dynamically Stable Motions	1321
56.5.3 Generating Whole-Body Motions from Operational Point Motions	1322
56.5.4 Generating Motions when in Contact with an Object	1324
56.6 Communication	1325
56.6.1 Expressive Morphology and Behavior	1325
56.6.2 Interpreting Human Expression	1327
56.6.3 Alternative Models for Human-Robot Communication	1329
56.7 Conclusions and Further Reading	1329
References	1329

56.1 Why Humanoids?

Throughout history, the human body and mind have inspired artists, engineers, and scientists. The field of humanoid robotics focuses on the creation of robots that are directly inspired by human capabilities (see

Chap. 60, *Biologically Inspired Robots*). These robots usually share similar kinematics to humans, as well as similar sensing and behavior. The motivations that have driven the development of humanoid robots vary widely.

For example, humanoid robots have been developed as general-purpose mechanical workers, as entertainers, and as test-beds for theories from neuroscience and experimental psychology [56.1–3].

56.1.1 The Human Example

On a daily basis, humans perform important tasks that are well beyond the capabilities of current robots. Moreover, humans are generalists with the ability to perform a wide variety of distinct tasks. Roboticians would like to create robots with comparable versatility and skill. When automating a task that people perform, it is natural to consider the physical and intellectual mechanisms that enable a person to perform the task. Exactly what to borrow from the human example is controversial. The literal-minded approach of creating humanoid robots may not be the best way to achieve some human-like capabilities (see Chap. 54, *Domestic Robots*). For example, dishwashing machines bear little similarity to the manual dishwashing they replace.

56.1.2 The Pleasing Mirror

Humans are humanity’s favorite subject. A quick look at popular magazines, videos, and books should be enough to convince any alien observer that humanity is obsessed with itself. The nature of this obsession is not fully understood, but aspects of it have influenced the field of humanoid robotics.

Humans are social animals that generally like to observe and interact with one another [56.4]. Moreover, people are highly attuned to human characteristics, such as the sound of human voices and the appearance of human faces and body motion [56.5–7]. Infants show preferences for these types of stimuli at a very young age, and adults appear to use specialized mental resources when interpreting these stimuli. By mimicking human characteristics, humanoid robots can engage these same preferences and mental resources.

Humanity’s narcissism has been reflected in media as diverse as cave paintings, sculpture, mechanical toys, photographs, and computer animation. Artists have consistently attempted to portray people with the latest tools at their disposal. Robotics serves as a powerful new medium that enables the creation of artifacts that operate within the real world and exhibit both human form and behavior [56.8].

Popular works of fiction have frequently included influential portrayals of humanoid robots and manmade humanoid creatures. For example, Karel Čapek’s sci-

ence fiction play Rossum’s Universal Robots (R.U.R.) is centered around the story of artificial people created in a factory [56.9]. This play from 1920 is widely believed to have popularized the term robot. Many other works have included explicit representations of humanoid robots, such as the robot Maria in Fritz Lang’s 1927 film *Metropolis* [56.10], and the thoughtful portrayal of humanoid robotics by Isaac Asimov in works such as *The Caves of Steel* from 1954 [56.11]. The long history of humanoid robots in science fiction has influenced generations of researchers, as well as the general public, and serves as further evidence that people are drawn to the idea of humanoid robots.

56.1.3 Understanding Intelligence

Many researchers in the humanoid robotics community see humanoid robots as a tool with which to better understand humans [56.3, 12]. Humanoid robots offer an avenue to test understanding through construction (*synthesis*), and thereby complement the careful *analysis* provided by researchers in disciplines such as cognitive science.

Researchers have sought to better emulate human intelligence using humanoid robotics [56.13]. Scientists, developmental psychologists, and linguists have found strong links between the human body and human cognition [56.14]. By being embodied in a manner similar to humans, and situated within human environments, humanoid robots may be able to exploit similar mechanisms for artificial intelligence (AI). Researchers are also attempting to find methods that will enable robots to develop autonomously in a manner akin to human infants [56.15]. Some of these researchers use humanoid robots that can physically explore the world in a manner similar to humans [56.16].

56.1.4 Interfacing with the Human World

Environments built for humans have been designed to accommodate human form and behavior [56.17, 18]. Many important everyday objects fit in a person’s hand and are light enough to be transported conveniently by a person. Human tools match human dexterity. Doors tend to be a convenient size for people to walk through. Tables and desks are at a height that is well matched to the human body and senses. Humanoid robots can potentially take advantage of these same accommodations, thereby simplifying tasks and avoiding the need to alter the environment for the robot [56.19]. For example, humanoid robots and people could potentially collaborate with one another in the same space using the same



Fig. 56.1 The humanoid robot HRP-1S driving a backhoe (Courtesy of Kawasaki Heavy Industries, Tokyu Construction and AIST). The robot can be teleoperated by a human operator to control the backhoe remotely. The same robot could potentially interface with many different unmodified machines

tools [56.20]. Humanoid robots can also interface with machinery that does not include drive-by-wire controls, as shown by the teleoperated robot in the cockpit of a backhoe in Fig. 56.1 [56.21].

Mobility serves as another example. It is very difficult to create a tall wheeled robot with a small footprint that is capable of traversing stairs and moving over rough terrain. Robots with legs and human-like behavior could potentially traverse the same environments that humans traverse, such as the industrial plant shown in Fig. 56.2, which has stairs and handrails designed for human use [56.23]. In addition to mobility advantages, legs have the potential to help in other ways. For example, legs could enable a humanoid robot to



Fig. 56.2 HRP-1 operating in a mockup of an industrial plant (Courtesy of Mitsubishi Heavy Industries)



Fig. 56.3 The humanoid robot HRP-2 dancing with a human [56.22]. The human is a master of a traditional Japanese dance whose dancing was recorded by a motion-capture system, and transformed for use by the robot

change its posture in order to lean into something, pull with the weight of its body, or crawl under an obstacle [56.24, 25].

56.1.5 Interfacing with People

People are accustomed to working with other people. Many types of communication rely on human form and behavior. Some types of natural gestures and expression involve subtle movements in the hands and face (Chap. 58, *Social Robots that Interact with People*). People can interpret eye gaze and facial expressions without training. Humanoid robots can potentially simplify and enhance human–robot interaction by taking advantage of the communications channels that already exist between people.

Similarly, people already have the ability to perform many desirable tasks. This task knowledge may



Fig. 56.4 Actroid (Courtesy of Kokoro), an android designed for entertainment, telepresence, and media roles

be more readily transferred to humanoid robots than to a robot with a drastically different body. This is especially true of cultural actions centered around the human form (Fig. 56.3).

56.1.6 Entertainment, Culture, and Surrogates

Humanoid robots are inherently appropriate for some applications. For example, many potential forms of entertainment, such as theater, theme parks, and adult companionship, would rely on a robot that closely re-

sembles a human, see Fig. 56.4. For a humanoid robot to be an improvement over a wax figure or an animatronic historical character, it must be realistic in form and function.

People may one day wish to have robots that can serve as an avatar for telepresence, model clothing, test ergonomics, or serve other surrogate roles that fundamentally depend on the robot's similarity to a person. Along these lines, robotic prosthetics have a close relationship to humanoid robotics, since they seek to directly replace parts of the human body in form and function (Chap. 53, *Health Care and Rehabilitation Robotics*).

56.2 History and Overview

There is a long history of mechanical systems with human form that perform human-like movements. For example, Al-Jazari designed a humanoid automaton in the 13th century [56.27], Leonardo da Vinci designed a humanoid automaton in the late 15th century [56.28],

and in Japan there is a tradition of creating mechanical dolls called *Karakuri ningyo* that dates back to at least the 18th century [56.29]. In the 20th century, animatronics became an attraction at theme parks. For example, in 1967 Disneyland opened its *Pirate's of the Caribbean* ride [56.30], which featured animatronic pirates that play back human-like movements synchronized with audio. Although programmable, these humanoid animatronic systems moved in a fixed open-loop fashion without sensing their environment.

In the second half of the 20th century, advances in digital computing enabled researchers to incorporate significant computation into their robots for sensing, control, and actuation. Many roboticists developed isolated systems for sensing, locomotion, and manipulation that were inspired by human capabilities. However, the first humanoid robot to integrate all of these functions and capture widespread attention was WABOT-1, developed by Ichiro Kato et al. at Waseda University in Japan in 1973 (Fig. 56.5).

The WABOT robots integrated functions that have been under constant elaboration since: visual object recognition, speech generation, speech recognition, bimanual object manipulation, and bipedal walking. WABOT-2's ability to play a piano, publicized at the Tsukuba Science Expo in 1985, stimulated significant public interest.

In 1986, Honda began a confidential project to create a humanoid biped. Honda grew interested in humanoids, perhaps seeing in them devices of complexity comparable to cars with the potential to become high-volume consumer products one day. In 1996, Honda unveiled the Honda Humanoid P2, the result of this confidential project. P2 was the first full-scale humanoid capable of

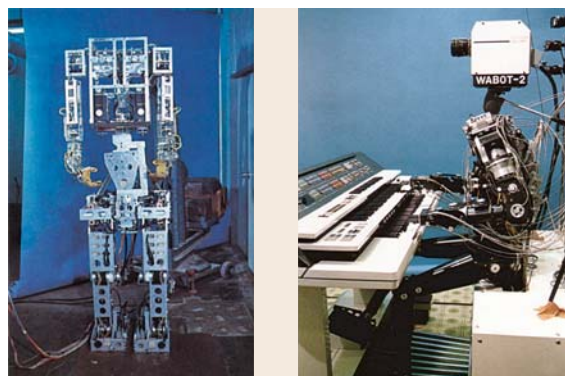


Fig. 56.5 WABOT-1 (1973) and WABOT-2 (1984) (Courtesy of Humanoid Robotics Institute, Waseda University)



Fig. 56.6 Honda P2 (180 cm tall, 210 kg), P3 (160 cm, 130 kg), and Asimo (120 cm, 43 kg) [56.26]. (Images courtesy of Honda)

stable bipedal walking with onboard power and processing. Successive designs reduced its weight and improved performance (see Fig. 56.6). Compared to humanoids built by academic laboratories and small manufacturers, the Honda humanoids were a leap forward in sturdiness, using specially cast lightweight high-rigidity mechanical links, and harmonic drives with high torque capacity.

In parallel with these developments, the decade-long *Cog project* began in 1993 at the MIT Artificial Intelligence laboratory in the USA with the intention of creating a humanoid robot that would, *learn to 'think' by building on its bodily experiences to accomplish progressively more abstract tasks* [56.13]. This project gave rise to an upper-body humanoid robot whose design was heavily inspired by the biological and cognitive sciences. Since the inception of the Cog project, many humanoid robotics projects with similar objectives have been initiated, and communities focused on developmental robotics, autonomous mental development (AMD [56.31]), and epigenetic robotics have emerged [56.32].

As of the early 21st century, many companies and academic researchers have become involved with humanoid robots, and there are numerous humanoid robots across the world with distinctive features.

56.2.1 Different Forms

Today, humanoid robots come in a variety of shapes and sizes that emulate different aspects of human form and behavior (Fig. 56.7). As discussed, the motivations that have driven the development of humanoid robots vary widely. These diverse motivations have lead to a vari-



Fig. 56.7 Kismet is an example of a humanoid head for social interaction



Fig. 56.8 The NASA Robonaut consists of an upper body placed on a wheeled mobile base

ety of humanoid robots that selectively emphasize some human characteristics, while deviating from others.

One of the most noticeable axes of variation in humanoid robots is the presence or absence of body parts. Some humanoid robots have focused solely on the head and face, others have a head with two arms mounted to a stationary torso, or a torso with wheels (see, for example, Fig. 56.8), and still others have an articulate and expressive face with arms, legs, and a torso. Clearly, this variation in form impacts the ways in which the robot can be used, especially in terms of mobility, manipulation, whole-body activities, and human–robot interaction.

56.2.2 Different Degrees of Freedom

Humanoid robots also tend to emulate some degrees of freedom in the human body, while ignoring others. Humanoid robots focusing on facial expressivity often incorporate actuated degrees of freedom in the face to generate facial expressions akin to those that humans can generate with their facial muscles. Likewise, the upper body of humanoid robots usually includes two arms, each with a one-degree-of-freedom (one-DOF) rotary joint at the elbow and a three-DOF rotary joint for the shoulder, but rarely attempt to emulate the human shoulder's ability to translate or the flexibility of the human spine [56.33, 34].

In general, humanoid robots tend to have a large number of degrees of freedom and a kinematic structure that may not be amenable to closed-form analysis due to redundancy and the lack of a closed-form inverse. This is in contrast to traditional industrial manipulators that are often engineered to have minimal redundancy (six DOFs) and more easily analyzed kinematic structures.

56.2.3 Different Sensors

Humanoid robots have made use of a variety of sensors including cameras, laser range finders, microphone arrays, lavalier microphones, and pressure sensors. Some researchers choose to emulate human sensing by selecting sensors with clear human analogs and mounting these sensors on the humanoid robot in a manner that mimics the placement of human sensory organs. As discussed in Sect. 56.6, this is perhaps most evident in the use of cameras. Two to four cameras are often mounted within the head of a humanoid robot with a configuration similar to human eyes.

The justifications for this bias towards human-like sensing include the impact of sensing on natural human-robot interaction, the proven ability of the human senses to support human behavior, and aesthetics. For example, with respect to human-robot interaction, nonexperts can sometimes interpret the functioning and implications of a human-like sensor, such as a camera, more easily. Similarly, if a robot senses infrared or ultraviolet radiation,

the robot will see a different world than the human. With respect to behavior, placement of sensors on the head of the robot allows the robot to sense the world from a vantage point that is similar to that of a human, which can be important for finding objects that are sitting on a desk or table.

Prominent humanoid robots have added additional sensors without human analogs. For example, Kismet used a camera mounted in its forehead to augment the two cameras in its servoed eyes, which simplified common tasks such as tracking faces. Similarly, versions of Asimo have used a camera mounted on its lower torso that looks down at the floor in order to simplify obstacle detection and navigation during locomotion.

56.2.4 Other Dimensions of Variation

Other significant forms of variation include the size of the robot, the method of actuation, the extent to which the robot attempts to appear like a human, and the activities the robot performs.

56.3 Locomotion

Bipedal walking is a key research topic in humanoid robotics (see also Chap. 16, *Legged Robots*, for a review of this topic in the context of locomotion in general). Legged locomotion is a challenging area of robotics research, and bipedal humanoid locomotion is especially challenging. Some small humanoid robots are able to achieve statically stable gaits by having large feet and a low center of mass, but large humanoids with a human-like weight distribution and body dimensions

typically need to balance dynamically when walking bipedally.

56.3.1 Bipedal Locomotion

Currently the dominant methods for bipedal legged locomotion with humanoids make use of the zero-moment point (ZMP) criterion to ensure that the robot does not fall over [56.35]. As discussed in detail in Chap. 16, con-

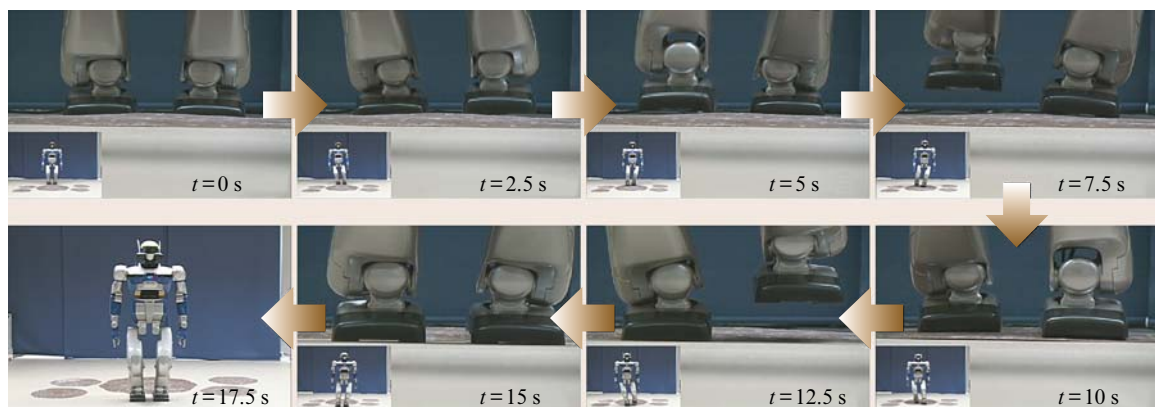


Fig. 56.9 HRP-2 walks on a slightly uneven surface



Fig. 56.10 These robots from Delft, MIT and Cornell (left to right) are designed to exploit their natural dynamics when walking [56.36]. (Image courtesy of Steven H. Collins)

trol of the robot's body such that the **ZMP** sits within the support polygon of the robot's foot ensures that the foot remains planted on the ground, assuming that friction is high enough to avoid slipping. The **ZMP** can be used to plan motion patterns that make the robot dynamically stable while walking.

Controllers based on the **ZMP** criterion try to follow a planned sequence of contact states and are often unable to change the landing positions in real time in response to lost contact. Current **ZMP**-based bipedal walking algorithms have difficulty handling unexpected perturbations, such as might be encountered with uneven natural terrain (Fig. 56.9). Robots using **ZMP** differ from human locomotion in significant ways. For example, unlike people, robots using **ZMP** typically do not exploit the natural dynamics of their legs, or control the impedance of their joints.

Augmenting **ZMP**-based control is currently an active area of research. As will be discussed in Sect. 56.5 on whole-body activities researchers are working to integrate manipulation and bipedal locomotion. For example, when a robot walks while grasping a handrail, the contact could potentially increase the stability of the robot, but the **ZMP** criterion does not easily generalize

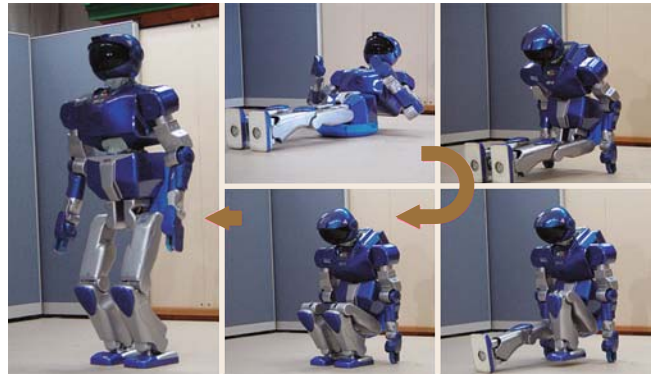


Fig. 56.12 The humanoid robot HRP-2P getting up from a lying-down position

to this task. So far, a generic and rigorous new criterion has not been established.

As an example of an alternative mode of bipedal locomotion, some running robots have used controllers based on an inverted pendulum model to achieve stable gaits. These methods change the landing positions to keep the robot dynamically stable [56.37]. More recently, researchers have begun to use the principles of bipedal passive-dynamic walkers to develop powered bipedal walkers that walk with high efficiency in a human-like way by exploiting natural dynamics (Fig. 56.10 [56.36]).

56.3.2 Falling Down

A human-scale robot should expect to fall from time to time in realistic conditions. A humanoid robot may fall down due to a large disturbance even if the motion is planned carefully and a sophisticated feedback controller is applied to the robot. In this event, the robot could be damaged significantly during a fall, and could also damage the environment or injure people who are nearby. An important area of research is how to control the robot's fall in order to gracefully recover or minimize damage. The Sony **QRIO** can control its falling motions in order to reduce the impact of touch down [56.38], al-



Fig. 56.11 Example of controlled falling-down motion

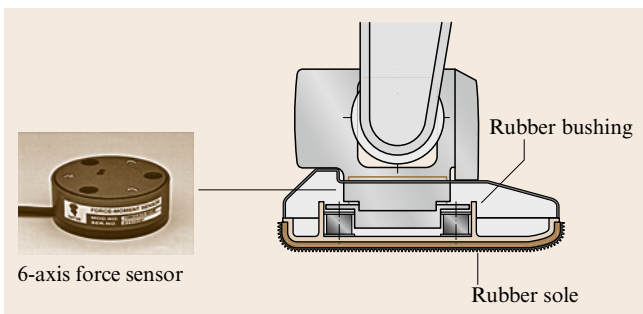


Fig. 56.13 Example of a humanoid foot structure for legged locomotion that uses compliance and force/torque sensing



Fig. 56.14 Asimo and artificial landmarks on the floor

though it is of a relatively small size (which simplifies the problem). *Fujiwara et al.* developed a falling motion controller for a human-size humanoid robot that is falling backwards [56.39]. Figure 56.11 shows an example of a controlled falling motion. The general problem is still very much an active area of research. Similarly,

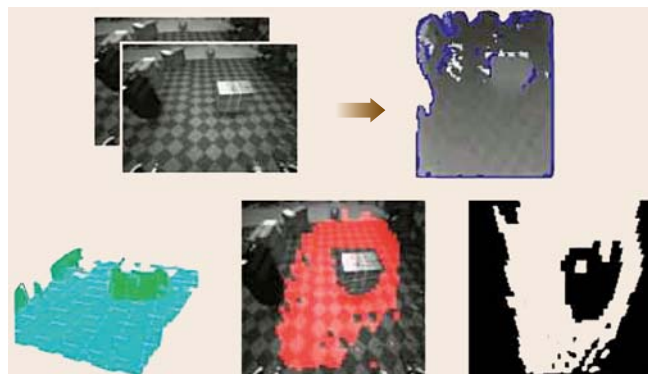


Fig. 56.15 Plane segment finder for detecting traversable floor area

there is also the issue of getting back up again [56.40] (Fig. 56.12).

56.3.3 Sensing for Balance

Bipedal walking needs to be robust to unexpected disturbances encountered during the execution of planned walking patterns. In these situations, walking can sometimes be stabilized with feedback control and appropriate sensing. Many humanoid robots, such as Honda's Asimo, make use of accelerometers, gyroscopes, and six-axis force/torque sensors to provide feedback to the robot during locomotion.

Force/torque sensors have long been applied to manipulators for the implementation of force control, but force/torque sensors with sufficient robustness to handle foot impact for a full-size humanoid robot are relatively new. When the foot of the robot touches down, the foot receives an impact which can disturb its walking. This impact can be rather large, especially when the robot is walking quickly. Some feet now incorporate a spring and damper mechanism as shown in Fig. 56.13 in order to mitigate these problems. As with many other aspects of bipedal humanoid locomotion, foot design is currently an open problem.

56.3.4 Localization and Obstacle Detection

In order for a humanoid robot to walk in unmodeled environments, localization and obstacle detection are essential. Wheeled robots encounter similar issues while navigating, but full bipedal humanoids have more-specialized requirements. For example, bipedal humanoids have the ability to control contact with the world through their highly articulate legs.

Artificial landmarks can simplify localization. As shown in Fig. 56.14, Honda's Asimo uses a camera mounted on its lower torso that looks down at the floor to find artificial markers for position correction [56.41]. Accurate positioning is important for long-distance navigation and stair climbing, since slippage usually occurs while walking and accumulated positional and directional errors can lead to severe failures.

Obstacle avoidance is also an important function for locomotion. Disparity images generated by stereo vision have been utilized for this purpose. For example, the plane segment finder [56.42] developed by *Okada et al.* helps detect traversable areas. Figure 56.15 shows the result of detecting clear areas of the floor plane appropriate for gait generation.

Humanoids require a great deal of computation due to the need for sophisticated sensing and con-

trol. Customized computational hardware may help mitigate this problem. For example, Sony's humanoid robot **QRIO** is equipped with a field-programmable gate array (FPGA) to generate disparity maps in

real time from the stereo cameras. This real-time vision system has been used to detect floor areas, stair steps, and obstacles for navigation [56.43, 44].

56.4 Manipulation

Hands and arms are the main interfaces with which humans act on the world around them. Manipulation research within humanoid robotics typically focuses on the use of anthropomorphic arms, hands, and sensors to perform tasks that are commonly performed by people. Several chapters of the handbook relate to these goals, including Chap. 24 (*Visual Servoing and Visual Tracking*), Chap. 26 (*Motion for Manipulation Tasks*), and Chap. 28 (*Grasping*).

56.4.1 The Arm and Hand

The kinematics of humanoid robot arms emulate the human arm, which can be approximated by seven degrees of freedom (DOFs), with three at the shoulder,

one at the elbow, and three at the wrist. The use of seven DOFs results in a redundant degree of freedom with respect to the six-DOF pose of the hand. To reduce mechanical complexity, humanoid robot arms sometimes have fewer than seven DOFs, for example, ARMAR-III and Justin have seven-DOF arms, Cog and Domo have six-DOF arms, and Asimo has five-DOF arms (Fig. 56.16) [56.45, 46].

Humanoid robot hands tend to vary more in their design (see Chap. 15, *Robot Hands*). The human hand is highly complex with over 20 DOFs (i. e., approximately four DOFs per finger and a five-DOF thumb) in a very compact space with a compliant exterior, dense tactile sensing, and muscular control. If a robot hand is to be mounted on a robot arm, there are additional constraints in terms of the mass of the robot hand, since the hand sits at the end of the arm and must be efficiently moved in



Fig. 56.16 The humanoid robot Justin has two seven-DOF torque-controlled arms (DLR-Lightweight-Robot-III), and two 12-DOF hands (DLR-Hand-II). Justin's body is larger than a human's



Fig. 56.17 3-D object recognition by HRP-2 using versatile volumetric vision

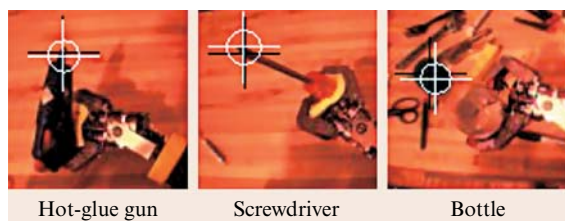


Fig. 56.18 Using visual motion, Domo detects (*white cross*) the tips of tool-like objects it is rigidly grasping with error (*white circle*) comparable to the error achieved with hand labels (*black cross*)

space. Researchers have approximated the human hand with varying levels of accuracy, including the ACT hand, the 20-DOF Shadow Hand, the 12-DOF DLR-Hand-II, the 11-DOF Robonaut hand, the two-DOF Cog hand, and the one-DOF Asimo hand [56.47–50].

The ACT hand is an excellent representative of the high-fidelity end of the spectrum, since it emulates the bone structure, inertial properties, and actuation of the human hand in addition to the kinematics. Humanoid robot hands often include passive degrees of freedom. For example, one DOF of the two-DOF Cog hand controlled a multijointed power grasp, while the other controlled a multijointed two-fingered precision grasp. Studies indicate that many human grasps can be approximated with two degrees of freedom [56.53], so simplified hands may be sufficient to emulate a variety of human manipulation activities.

56.4.2 Sensing for Manipulation

Model-Based Vision

A common approach to visual perception for humanoid robots is real-time three-dimensional (3-D) object recog-

nition and six-DOF pose estimation using models of the objects to be manipulated (see Chap. 23, *3-D Vision and Recognition*).

An example of this approach is provided by the versatile volumetric vision (VVV) system, which is an edge-based 3-D vision system developed by Tomita et al. [56.54]. The VVV system has been utilized to find and grasp objects during everyday manipulation tasks. Figure 56.17 shows a demonstration of the HRP-2 recognizing a beverage can on a table so that it can pick up the can and throw it into a trash can. This integrated system was demonstrated at Aichi EXPO 2005, where it enabled a human operator to control the HRP-2 humanoid robot in a semi-autonomous fashion with reduced effort [56.55]. The VVV system has also helped HRP-2 carry a table in cooperation with a human [56.56].

Other robots have used similar approaches. For example, Robonaut from the National Aeronautics and Space Administration (NASA) Johnson Space Center (JSC) has used model-based vision to perform everyday manipulation tasks, such as tightening lug nuts on a wheel [56.57].

Feature-Based Vision

Another approach to visual perception for humanoid robots is feature-based vision. Encoding tasks in terms of task-relevant features, such as the tip of a tool or the contact surface of a hand, offers an alternative to approaches that use detailed 3-D models of objects. In order to generalize a task across different objects, only the task-relevant features need to be detected and mapped.

An example of this approach is provided by the humanoid robot Domo, which uses task-relevant features to perform everyday tasks such as pouring, stirring, and brushing. Domo detects features such as the opening of a container or the tip of a tool (Fig. 56.18) and then visually serveses these features with respect to one another in order to perform a task. Once the objects are in contact with one another, Domo uses force sensing and compliance to simplify tasks such as regrasping, inserting, and placing objects [56.52, 58].

Active Perception

Through action, robots can simplify perception. Humanoid robots have used this approach in ways that are reminiscent of human behavior. For example, a humanoid robot can reach out into the world to physically sense its surroundings (Fig. 56.19), or induce visual motion through physical contact so as to better estimate the extent of manipulable objects [56.59]. Similarly, a hu-

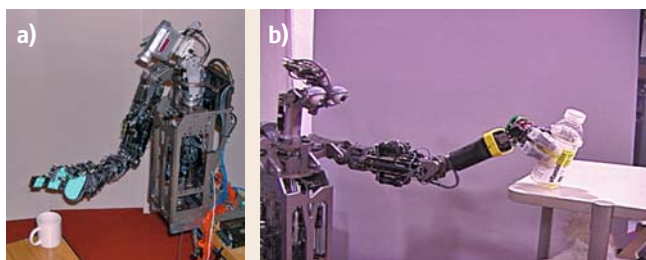


Fig. 56.19a,b The humanoid robots Obrero (a) and Domo (b) use passive compliance and force control to safely reach out into the world. Obrero haptically grasps an object [56.51]. Domo physically finds the shelf, and uses force control to let objects settle into place [56.52]

manoid robot can select postures that enable it to more easily view visual features that are relevant to the current task.

Force, Torque, and Tactile Sensing

When in contact with the world, force, torque, and tactile sensing are especially valuable (see Chap. 19, *Force and Tactile Sensors*), for example, as shown in the section on whole-body activities, force/torque sensing in the arms can be used to help a humanoid robot stay balanced while lifting or pushing an object for which an accurate model with friction coefficients and mass does not exist (Figs. 56.36 and 56.23). Likewise, many of the capabilities of the humanoid robots Cog, Domo, and Obrero depend on force, torque, and tactile sensing. As another example, the humanoid robot Justin from DLR uses torque sensing at all of its joints to perform Cartesian impedance control, which can implement virtual springs to coordinate bimanual manipulation of large objects (Fig. 56.16) [56.46].

The extent to which passive compliance and torque control should be integrated into humanoid robots is still an open issue (see Chap. 7, *Force Control*). Many humanoid robots have stiff joints that are well suited to position control, or perform force control using feedback from a force/torque sensor near the point of contact, such as the wrist or foot. However, even when a human attempts to maintain the position of her hand, the joints of her arm continue to exhibit significant compliance [56.60, 61] (Chap. 13, *Robots with Flexible Elements*).

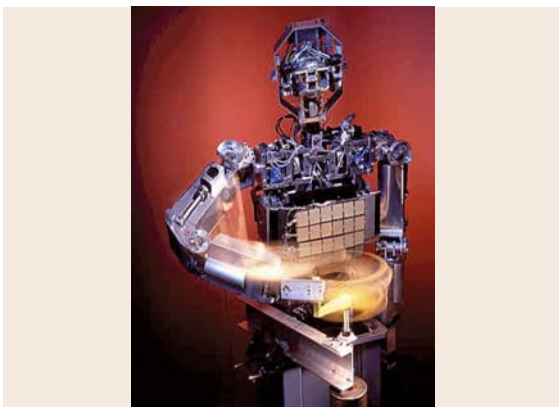


Fig. 56.20 The humanoid robot Cog used neural oscillators in conjunction with compliant force-controlled arms to perform a variety of everyday tasks with human tools, such as crank turning, hammering, and sawing. Photo by Sam Ogden

56.4.3 Rhythmic Manipulation

Many everyday tasks performed by humans involve rhythmic motions rather than discrete motions [56.62].

A good example of rhythmic humanoid manipulation was provided by Williamson, who demonstrated that central pattern generators (CPGs) in conjunction with compliant force-controlled arms can enable a humanoid robot to perform many everyday rhythmic manipulation tasks in a manner similar to humans (Fig. 56.20). His approach enabled the humanoid robot Cog to perform a variety of tasks without motion planning or model-based control. These tasks included hammering nails, sawing through wood, turning a crank at various orientations, turning cranks bimanually, playing with a slinky toy bimanually, and playing a snare drum [56.63]. The CPGs entrained on the natural dynamics of the robot's arm coupled to the environment. The output of the CPGs were used to control the set point of virtual springs in the joints, which were actuated with series elastic actuators (SEAs) [56.64].

Rhythmic motion can also be used as a basis for learning human-like manipulation tasks. For example, The humanoid robot DB has learned to perform rhythmic manipulation tasks, such as drumming, from human demonstration [56.65].

56.4.4 Cooperative Manipulation

Humanoid robots have potential advantages when working with humans to perform tasks cooperatively [56.20] (see Chap. 57, *Safety for Physical Human-Robot Interaction*). Humans typically have extensive experience cooperating with other humans to perform manual tasks. Humanoid robots have the potential to simplify this

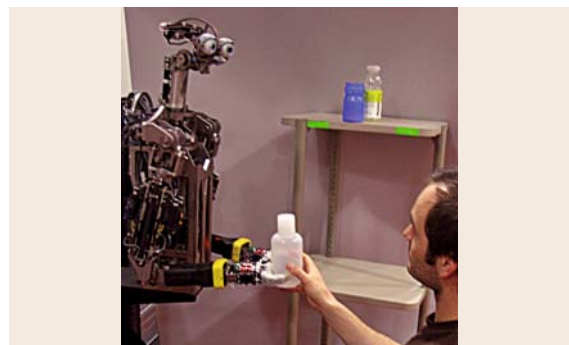


Fig. 56.21 Domo works with a human to place objects on a shelf [56.52]

form of human–robot interaction by closely matching a human counterpart during cooperative activities. For example, this can reduce the need for a human cooperator to learn how to interact with novel kinematic structures and sensing, and enable the robot and the human to use the same tools. Robonaut has been teleoperated to perform an insertion task in cooperation with a human [56.66]. Work with Domo has shown that people without prior experience readily interpret reaching cues and place an object in Domo’s hand in a favorable pose (Fig. 56.21) [56.52, 67]. Likewise, as shown in the section on whole-body activities (Sect. 56.6), humanoid robots have the potential to help a person move large objects (Fig. 56.36).

56.4.5 Learning and Development

Humans are very successful at learning new manipulation tasks, and human infants progressively develop sophisticated manipulation abilities over the course of several years. Researchers within humanoid robotics have attempted to emulate these aspects of human intelligence.

Learning by demonstration is a common way for people to learn from one another. Since humanoid robots have a similar form to humans, and have been designed to perform similar tasks, learning by demonstration may have advantages for skill acquisition [56.68, 69] (see Chap. 59, *Robot Programming by Demonstration*). The humanoid robot ARMAR has a framework for learning bimanual tasks from demonstration, which has enabled ARMAR to open a container with a screw-on lid [56.70]. Researchers have also sought to integrate learning by



Fig. 56.22 The humanoid robot DB learns to play air hockey against a human opponent through observation and practice. ©JST, ATR; Robot developed by SARCOS

demonstration with learning from exploration and motor primitives. For example, *Bentivegna* et al. have demonstrated methods that enable the humanoid robot DB to learn to play air hockey by observing a human opponent and practicing (Fig. 56.22) [56.71].

Biologically inspired approaches to humanoid manipulation are also common (see Chap. 62, *Neuro-robotics: From Vision to Action*). Before contact, the hand must move through free space to a target. Researchers have worked to enable robots to develop reaching behaviors using methods inspired from infant development [56.15]. Likewise, researchers such as *Platt* et al. have created methods that enable a humanoid robot to learn to grasp everyday objects in order to perform common tasks, such as grasping grocery items and placing them in a bag [56.72].

56.5 Whole-Body Activities

The two previous sections have focused on humanoid locomotion and humanoid manipulation independently. This section looks at mobile manipulation using a full humanoid robot with both arms and legs. Researchers wish to enable these humanoids to perform tasks such as lifting and carrying a box (as shown in Fig. 56.23), climbing a ladder, or even playing a sport. This section provides a brief glimpse into several approaches to this challenging problem.

Unlike the industrial manipulators described in Chap. 42 (*Industrial Robotics*), or a humanoid that is rigidly fixed in place, a bipedal humanoid must be controlled so as to remain balanced while manipulating an

object. Without actively maintaining stability, the robot could tip over and fall to the ground.

Moreover, a humanoid robot usually has many joints compared to a standard industrial manipulator. Humanoids are a kind of redundant robot as described in Chap. 11 (*Redundant Manipulators*). Consequently, there are many postures that can achieve a given pose of the robot’s hand. Out of the possible configurations of the robot’s body, many will result in the robot falling over.

Figure 56.24 shows an overview of several approaches to whole-body motion generation for humanoid robots. These methods decouple the problem

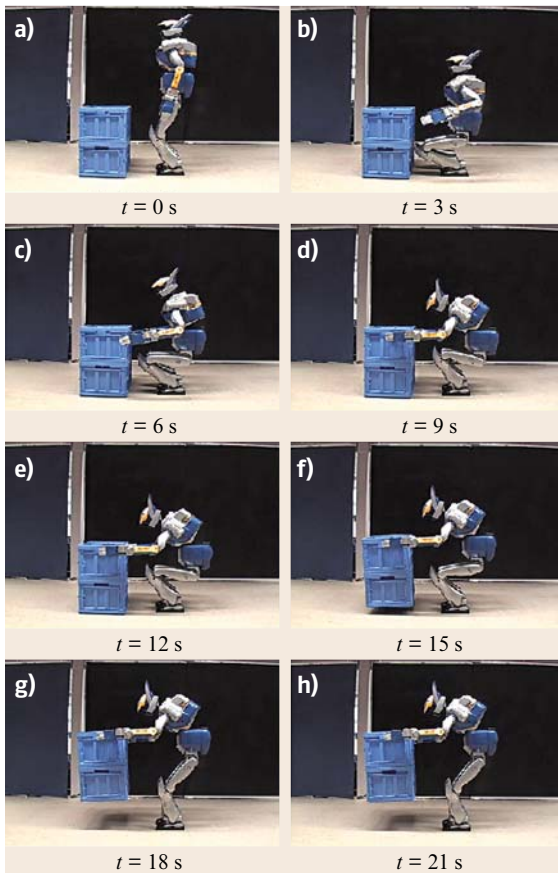


Fig. 56.23 Lifting an object while moving the waist to compensate for the load [56.73]

by computing a coarse movement and transforming it into a dynamically balanced movement. Then, during the whole-body activity, the humanoid uses a sensory

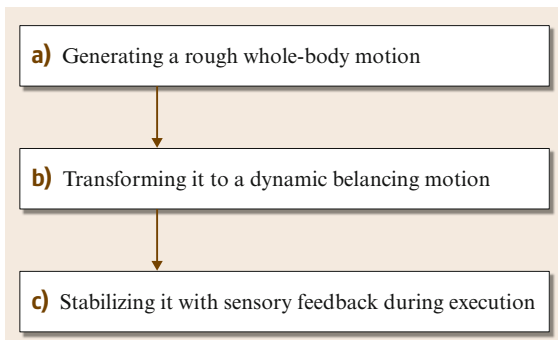


Fig. 56.24 Overview of motion-generation stages for a balancing robot

feedback control system to stabilize its motion. If the generated motion pattern were perfect and the state of the world could be exactly anticipated in advance, this feedback might be unnecessary. However, it is very difficult, if not impossible, to make a perfect motion pattern during real-world situations, since models of the robot and the environment inevitably have errors, for example, surface conditions, such as friction and compliance, can be very important, yet difficult to anticipate (Chap. 27, *Contact Modeling and Sliding Manipulation*).

This section focuses on the first two parts of the process shown in Fig. 56.24: coarse motion generation and the conversion of coarse motions into detailed motions that enable whole-body activities.

56.5.1 Coarse Whole-Body Motion

There are currently four prevalent ways to generate coarse whole-body motion:

1. Using a motion capture system
2. using a graphical user interface (GUI) offline
3. using a teleoperation interface in real time
4. using automatic motion planning

Using a Motion Capture System

Humanoid robots have a human-like shape and many researchers would like humanoid robots to perform tasks in a human-like way. Consequently, human motion is often used when generating the motion of a humanoid robot. Motion capture systems, such as those used by movie studios for special effects, are commonly used for recording human motion. Typically a human subject

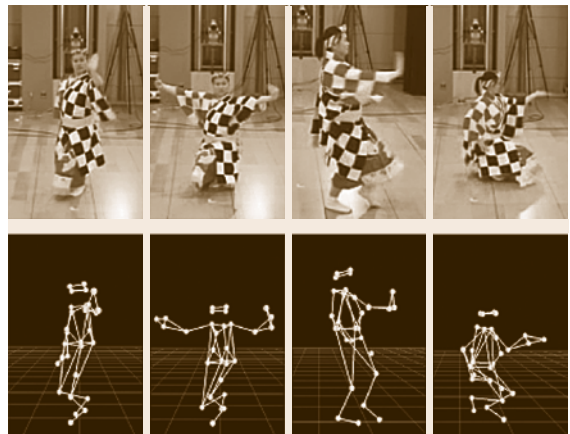


Fig. 56.25 An example of captured motion data for a Japanese folk dance [56.22]

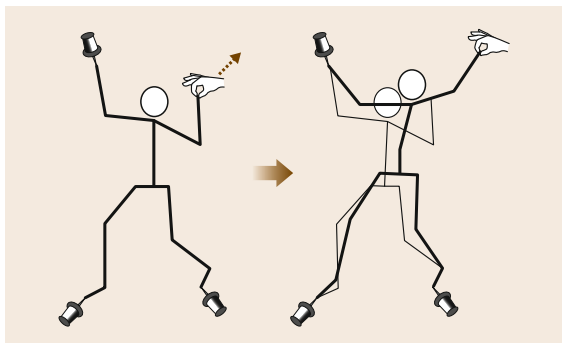


Fig. 56.26 The *pin-and-drag* interface for generating coarse movements [56.74]

performs actions while wearing easily detected markers on his or her body. The motion of these markers are recorded by cameras placed in the room, and software then infers the 3-D positions of these markers over time. Figure 56.25 shows an example: the captured motions of a woman performing a Japanese folk dance [56.22].

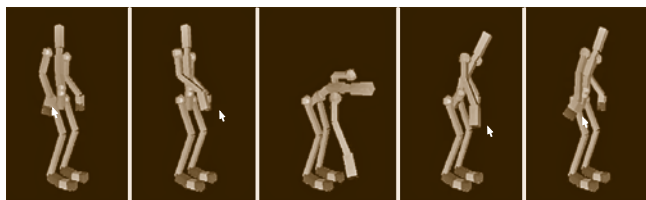


Fig. 56.27 An example of a whole-body motion generated using the pin-and-drag interface [56.74]

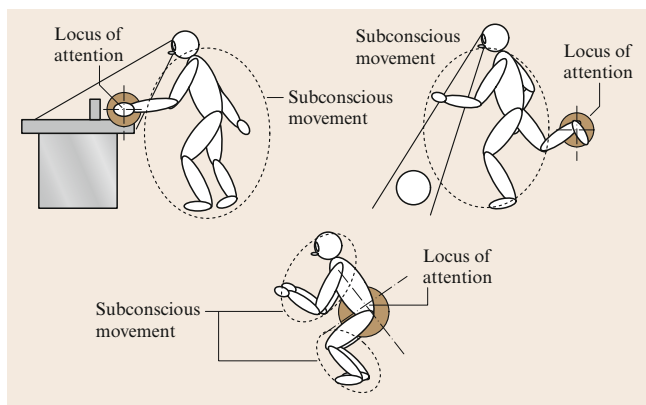


Fig. 56.28 While teleoperating a robot, the human operator can focus his or her attention on a small part of the humanoid robot. Automatic techniques can then take this coarse motion specification and automatically generate full motion specifications that meet important constraints, such as maintaining the robot's balance

This is a mature technology, so motions such as those shown in Fig. 56.25 can easily be obtained and then used as a reference for a humanoid's whole-body motion. Due to kinematic similarities, coarse kinematic motions can be transferred to a humanoid robot. However, due to dynamic differences, such as mass distributions and torque generation, captured motions may not be stable or feasible. Moreover, the task constraints and sensory feedback involved in a motion-captured activity may not be easily inferred, which can lead to robot motion that is similar in appearance, yet fails to achieve the goals of a motion.

Using a GUI Offline

Tools such as those used in character animation for computer graphics can also be used to design movements for humanoid robots. If the designer were forced to control each of the many degrees of freedom independently, the process would be tedious and inefficient. One approach to overcoming this issue is a *pin-and-drag GUI* interface, which can help the designer more efficiently control the motion of all the joints of a high degree of freedom humanoid [56.74]. Figure 56.26 outlines this interface. The user can pin links of the humanoid to desired locations. Then the software automatically generates natural movements of the humanoid's joints subject to these constraints, while the user drags other links. Figure 56.27 shows a whole-body movement generated by dragging the right hand up and down for four seconds while pinning the left hand, the toes, and the heels. This results in natural-looking movements for picking up an object. As with the motion capture system, this generated whole-body motion may not be directly applicable to a humanoid robot due to dynamics and modeling errors.

Using a Teleoperation Interface in Real Time

Given the complexity and potential instability of humanoid robots, every detail of a whole-body activity cannot be efficiently controlled by a human in real time. The teleoperation (Chap. 31, *Telerobotics*) of humanoid robots combines coarse motion control provided by a user with methods that generate effective full-body motions. Since teleoperation occurs in real time, the methods used to perform this transformation cannot rely on knowledge of future motion commands, and must be computationally efficient. This rules out the use of methods such as the autobalancer (described later), which requires an entire motion trajectory in order to globally optimize the conversion.



Fig. 56.29 An example whole-body motion generated by the RRT motion planning system [56.77]

Human motions are generated with a mixture of two kinds of motion: motions that require careful attention and motions that require little cognitive effort (Fig. 56.28). A similar division can be used when designing a teleoperation system for whole-body activities [56.75]. For example, in a manner similar to the previously discussed GUI interface, a human operator can control specific points on a humanoid robot's body in order to generate coarse motions. By reducing the aspects of the motion that the human operator must specify, he or she can control a humanoid robot to perform complex motions in real time using simple interfaces, such as joysticks or high-level commands [56.76].

Using Automatic Motion Planning

Motion capture, teleoperation, and GUI-based motion editing tools give the user interactive control over the robot's motions, but they can be time consuming

to use. Fast path-planning techniques such as rapidly exploring random trees (RRT) [56.79] can compute collision-free full-body motions for humanoid robots automatically [56.77] (see Chap. 5, *Motion Planning*, and Chap. 26, *Motion for Manipulation Tasks*). Given geometric models of the humanoid and the environment, an initial posture, and a goal posture, the planning system automatically searches for a collision-free whole-body motion that moves the humanoid from the initial posture to the goal posture (Fig. 56.29). This type of method can also search for statically stable postures that smoothly interpolate between the initial and goal posture, and explicitly include other constraints based on the dynamics.

56.5.2 Generating Dynamically Stable Motions

The methods presented in Sect. 56.5.1 can be useful when generating coarse whole-body motions for a humanoid robot. However, for some of these methods the motions generated will not take into account the dynamic stability of the robot, and may result in the robot falling over. This subsection presents two example techniques for converting coarse motions to dynamically-stable motions: the dynamics filter, and the autobalancer (other methods such as strict trunk motion computation used in the motion generation system for waseda bipedal humanoid (WABIAN) [56.80] are also possible).

Dynamics Filter

The dynamics filter proposed by Yamane and Nakamura [56.78] can convert a physically infeasible motion into a feasible one for a given humanoid. This consists of a controller and optimizer as shown in Fig. 56.30.

The controller calculates reference joint accelerations by using local and global feedback loops with respect to the given coarse motions and the current state of the humanoid model. The optimizer calculates joint accelerations that minimize the difference between the reference joint acceleration and the current joint acceleration under the given contact constraints, such as floor friction and contact force feasibility. Figure 56.31 shows motion-captured data of a human walking and the resulting, physically feasible motion produced by the dynamics filter. The motions look similar, but the contact conditions for the feet have been modified to be physically feasible.

Autobalancer

The autobalancer calculates all joint angles at every sample in time by solving a quadratic programming op-

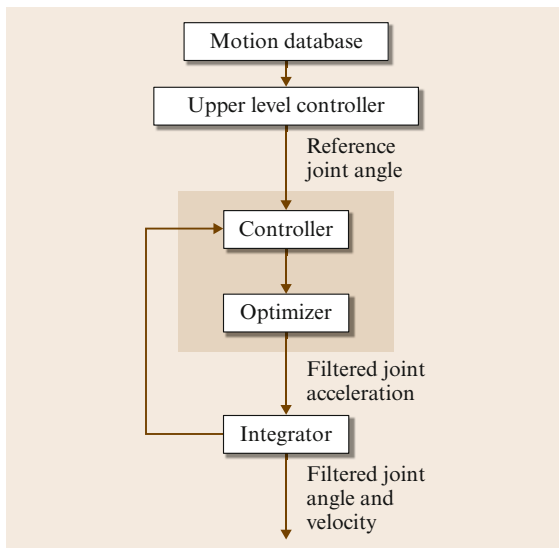


Fig. 56.30 Dynamics filter system diagram [56.78]

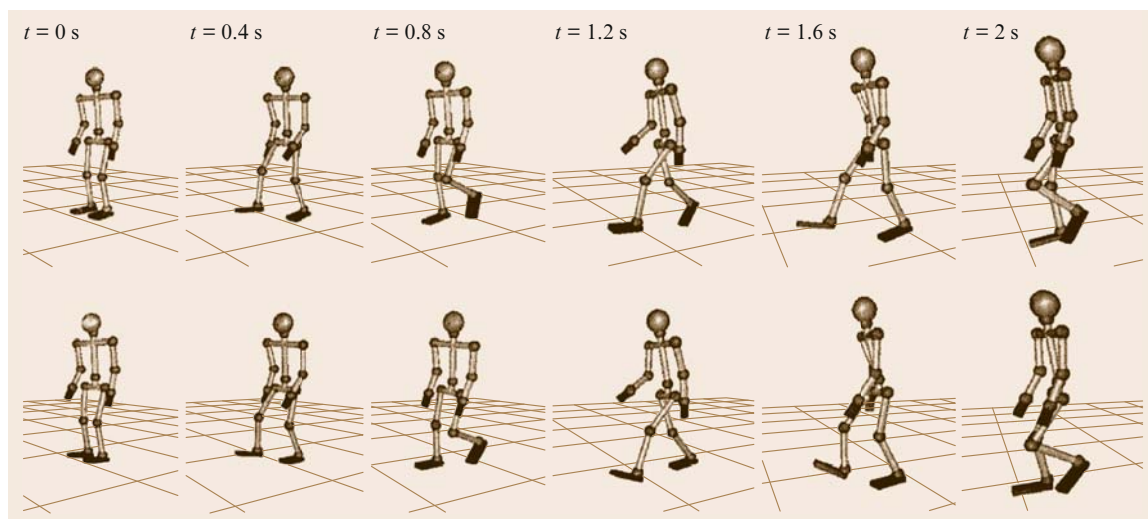


Fig. 56.31 An example of motions converted by the dynamics filter [56.78]; the top row shows the original coarse planned movements, and the lower row shows the modified movements that meet necessary constraints

timization problem in order to convert a given motion to a balanced one [56.81]. This method is not suitable for a motion in which dynamic balancing is dominant, such as during walking motions. However, this approach can be effective for a motion in which static balancing is dominant, such as when the humanoid is standing. The autobalancer can sometimes generate a walking motion if the robot can move slowly enough to minimize the influence of the dynamics while still maintaining stability [56.81].

The autobalancer calculates a whole-body motion based on the following concepts.

1. Fix the center of gravity (CoG) on the vertical axis which passes through a point in the support polygon of the humanoid;
2. Keep inertia moments around the CoG at acceptable values in order to satisfy the balancing conditions.

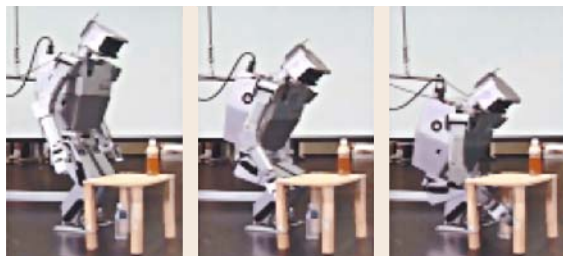


Fig. 56.32 The humanoid robot H7 executes a motion generated by the autobalancer [56.77]

Acceptable values for the moments are obtained by using constraints that keep the ZMP in the support polygon and keep the angular momentum generated by the moments close to zero over a given length of time. This problem can be handled as a quadratic programming optimization that minimizes errors between the given reference motion and the calculated motion given the constraints on the position of the CoG and the generated moments. The autobalancer solves this problem over every sampling period. Figure 56.32 shows the humanoid robot H7 executing a whole-body motion that resulted from using the autobalancer to transform the automatically planned coarse motions shown in Fig. 56.29 [56.77].

56.5.3 Generating Whole-Body Motions from Operational Point Motions

The example GUI and teleoperation interface described previously, enable a person to control the motions of particular points on a robot. Similarly, many automated control systems have been designed to control particular points on a robot, such as the end effector [56.82] (see Chap. 26, *Motion for Manipulation Tasks*). In order to control a real or simulated humanoid robot, these coarse motions must be converted to whole-body motions.

The computational efficiency and causality of this conversion is important. Teleoperation requires real-time methods, and human interfaces benefit from interactive control that provides real-time feedback.

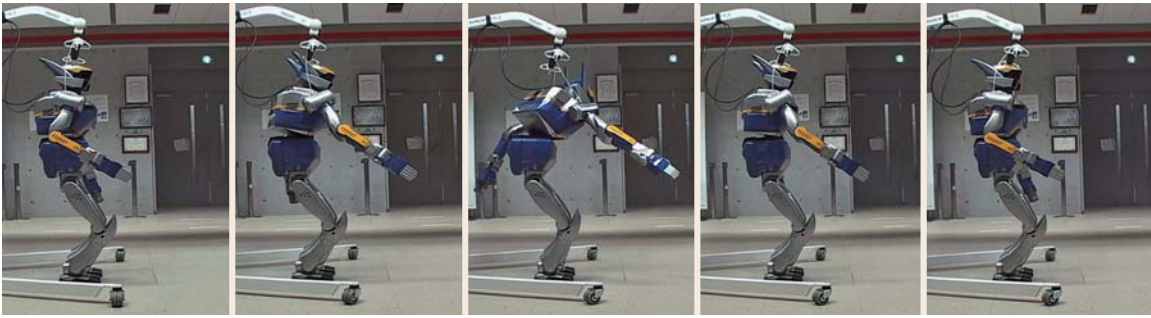


Fig. 56.33 Forward reaching motion with balance [56.75]

Likewise, the autonomous control of real robots with nontrivial dynamics requires real-time methods. Many approaches have been developed to generate whole-body motions from operational point motions in real time, two examples of which follow.

Resolved Momentum Control

Kajita et al. proposed a framework for whole-body control based on the linear and angular momentum of the entire robot [56.85]. Intuitively, one can think of the robot as a single rigid body whose linear and angular momentum is to be controlled. For example, the center of mass of the robot might be commanded to move at a fixed height above the ground. At each point in time, this framework uses least squares to find joint velocities that will achieve the desired linear and angular momentum of the robot. Elements of the momentum can also be left unspecified as free variables, which is often done in practice with elements of the angular momentum. In addition to elements of the momentum, resolved momentum control requires that desired velocities for the feet be specified.

Neo et al. adapted this framework to generate whole-body motions from operational point motions specified

by a human operator [56.75, 76]. The human operator can control the foot motion, the overall motion of the humanoid robot, or the motion of a hand at any time by switching the mode of operation. The whole motion-generation system then computes joint velocities that meet the desired motions and additional constraints.

Reaching tasks performed by the humanoid HRP-2 provide an example of this type of operation [56.86]. Figure 56.33 shows snapshots of the robot's movement with the operator controlling only the velocity of the right hand in order to perform a horizontal reaching motion. The robot automatically moves other joints, such as its torso, in order to stay balanced while the operator solely controls the hand.

Coordinating Constraints, Tasks, and Postures

Dynamic stability is very important during whole-body activities. However, humanoid motion is often subject to many simultaneous objectives that may be in competition with one another. *Sentis* and *Khatib* proposed a framework that enables a humanoid robot to perform motions that simultaneously meet prioritized objectives in real time [56.87], and *Park* and *Khatib*

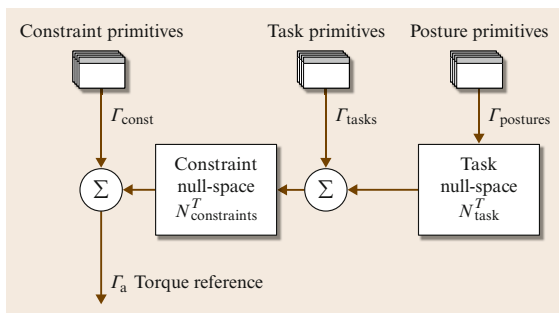


Fig. 56.34 A control hierarchy is established using null-space projections [56.83]

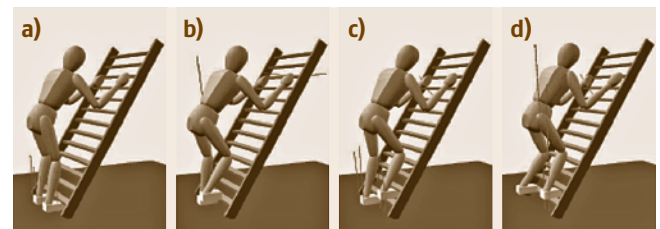


Fig. 56.35a–d Climbing a ladder. (a) The robot begins to climb; it has contacts on both hands and feet. (b) The right foot is then controlled to move up one step. (c) Next, the center of mass is controlled to move to the right in order to maintain balance with two hands and the right foot. (d) The left foot is then controlled to move up one step [56.84]

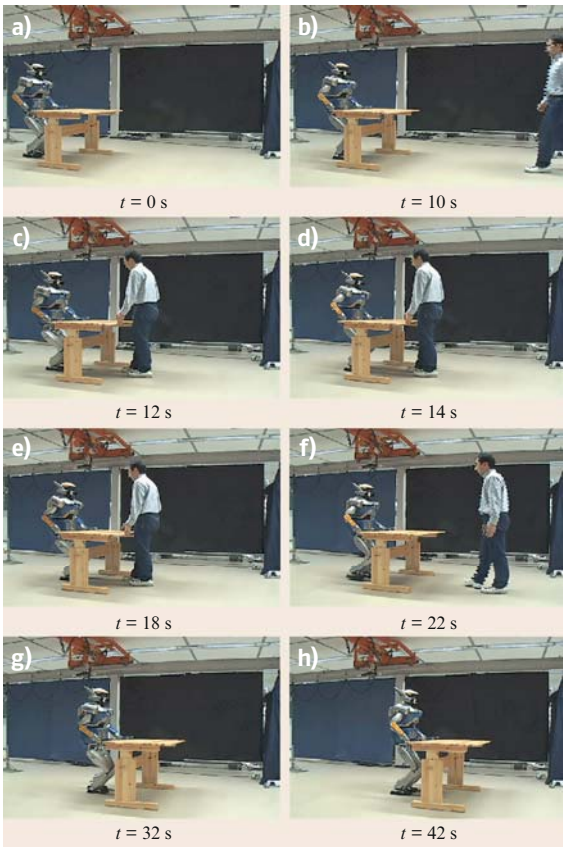


Fig. 56.36 Example of pushing manipulation and cooperation [56.88]

have augmented this approach to handle contact with the environment [56.84]. With these methods many controllers can be coordinated, and each controller can specify different objectives, such as a desired trajectory for a particular point on the robot.

This coordination of controllers uses three distinct control categories: constraint handling, operational

tasks, and postures. A control hierarchy handles conflicting scenarios among these three categories of controllers. Constraint primitives, such as joint limits, have the highest priority and the posture primitives have the lowest priority. One advantage of the posture primitives is that they can control redundant degrees of freedom in a consistent way.

To achieve this control hierarchy, operational tasks are projected into the constraint null space, and postures are projected into the task null space followed by the constraint null space (Fig. 56.34). Unlike resolved momentum control, which generates joint velocities, all of these controllers generate joint torques. When calculating these torques, the controllers use models of the dynamics of the robot and the environment.

A ladder-climbing behavior tested in simulation demonstrates this framework for control (Fig. 56.35). In this example, the desired trajectories for the center of mass, the hands, and the feet are specified in advance. When the simulation is run, the whole-body control system generates joint torques that seek to meet these coarse motion specifications, which results in the simulated robot climbing the ladder. While climbing, the simulated robot successfully resists unexpected disturbances.

Research into methods for the coordination of controllers for humanoid robots is an active area of research, for example, *Mansard et al.* have used a *stack of tasks* to organize prioritized controllers, including a visual servoing controller and a grasping controller, to enable a real HRP-2 robot to visually grasp a pink ball while walking [56.89].

56.5.4 Generating Motions when in Contact with an Object

Many approaches to whole-body motion generation assume that the robot is only in contact with the ground. When a humanoid robot's hands make contact with the environment, it can no longer maintain balance using

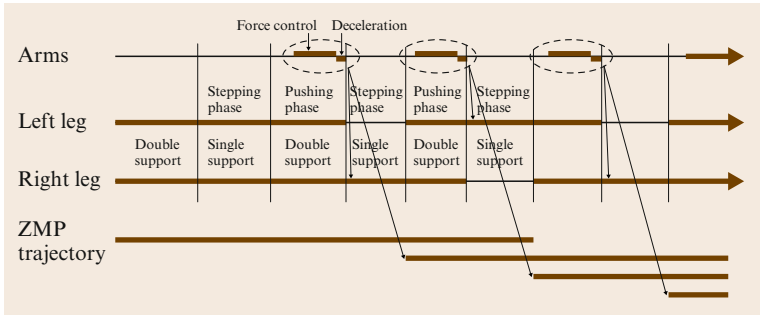


Fig. 56.37 Time chart of force-controlled pushing manipulation

the conventional **ZMP** property defined by the center of pressure of the supporting feet [56.90]. This leads to significant challenges for whole-body activities, especially since the properties of the environment with which the robot is making contact may not be known in advance.

Harada has introduced generalized **ZMP** (**GZMP**) as a method of handling some of these issues, such as the hand reaction forces generated from contact with the environment [56.91]. Researchers have developed methods that directly make use of the six-dimensional force/torque acting on the robot at the hands, which can be sensed with conventional force/torque sensors placed at the wrists [56.92]. Researchers have also developed specialized methods for generating stable robot motion while an object is being manipulated [56.88, 90, 93–95].

Carrying an Object

In a manner analogous to the previously described methods, coarse motions that do not consider the hand reaction forces can be modified [56.73, 90, 93, 96]. Figure 56.23 shows an experimental result of carrying an object that weighs 8 kg [56.73]. Based on measurements of the hand reaction force, the position of the waist is modified to compensate for the load and maintain stability.

Pushing an Object

As another example of using force sensing to adapt behavior, consider the problem of pushing a large object

placed on the floor. For such a task, if the gait pattern is determined before the robot actually moves, the robot may not stay balanced if the weight of the object or the friction coefficient between the object and the floor is different from the predicted values. To address this problem, the gait pattern can be adaptively changed depending on the output of a force sensor at the end of the arms in order to handle changes in the object's weight and the friction coefficient [56.88].

One approach to this activity is to have the robot push the object during the double support phase of walking, when both feet are in contact with the floor. A timing chart of this strategy for force-controlled pushing is shown in Fig. 56.37. During the pushing phase, the robot pushes the object while controlling the reaction forces applied at the hands. In the stepping phase, the step length is set to match the distance the object has been pushed in the pushing phase. The desired **ZMP** trajectory is recalculated and connected to the current **ZMP** trajectory. The trajectory of the center of gravity is also recalculated in order to realize this **ZMP** trajectory.

Figure 56.36 shows an experimental result for this approach [56.88]. In the experiment, the table weighs about 10 kg. Even though the motion of the table is disturbed externally during the experiment, the robot stays balanced by adaptively changing its gait pattern based on the measured forces.

56.6 Communication

Humans evaluate each others' state through body posture and movement. It is quite natural to extend this form of communication to include robots that share our morphology.

56.6.1 Expressive Morphology and Behavior

Humanoids can communicate with people through expressive morphology and behavior. As with people, humanoid robots integrate communicative and noncommunicative functionality. For example, the arms and hands of a robot can reach and grasp, but also point and gesture. Heads for humanoid robots are an especially important example of these overlapping roles, and have had an important impact on humanoid robotics and robotics in general [56.97].

The head of a humanoid robot has two main functions:

- To orient directional sensors as needed for the purposes of perception, while leaving the main body free to meet other constraints such as maintaining balance and gait. Cameras and sometimes microphones are usefully oriented in this way.
- To strike expressive poses, along with the rest of the body. Even if a robot head is not intended to be expressive, it will be interpreted as being so by humans – particularly as a cue to the robot's presumed locus of visual attention. It is also possible to deliberately engineer an approximate *face* that can be an important line of communication with humans (see Chap. 58, *Social Robots that Interact with People*).

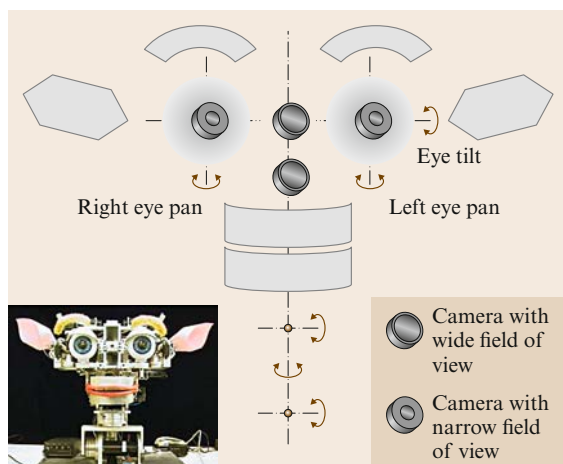


Fig. 56.38 On Kismet, foveal vision was implemented using cameras in the eyes, and peripheral vision used unobtrusive cameras on the head [56.97]. This achieved good expression of locus of attention, while simplifying the process of differentiating egomotion from motion of objects (since the head moved less frequently and more slowly than the eyes). This is an example of a partial decoupling of expressive and functional concerns, showing that many different levels of humanoid *fidelity* are possible

Locus of Attention

Eyes can be one of the most expressive components of a humanoid robot. For humans, eye movements are both expressive and important for sensing. Humanoid robots have the option to factor these two roles by moving eyes that are only for display, and using sensors placed elsewhere. Most humanoid robots, however, use head-mounted servoed cameras that play both expressive and sensory roles. These mechanisms exhibit different degrees of biological realism, for example, the Kismet head captured many of the expressive components of human eye movements, while having a non-human-like camera arrangement that simplified some forms of perception (Fig. 56.38).

Many humanoid robots use biologically inspired, foveated vision systems, which provide a wide field of view with low detail, combined with a narrow field of view with high detail (Fig. 56.39). With appropriate control strategies to fixate the narrow field of view on task-salient regions detected in the wide field of view, these robots achieve a practical compromise between resolution and field of view. Additionally, the configuration of the eyes communicates the robot's locus of attention in an intuitive way. Many systems use four cameras, with a narrow- and wide-angle camera for

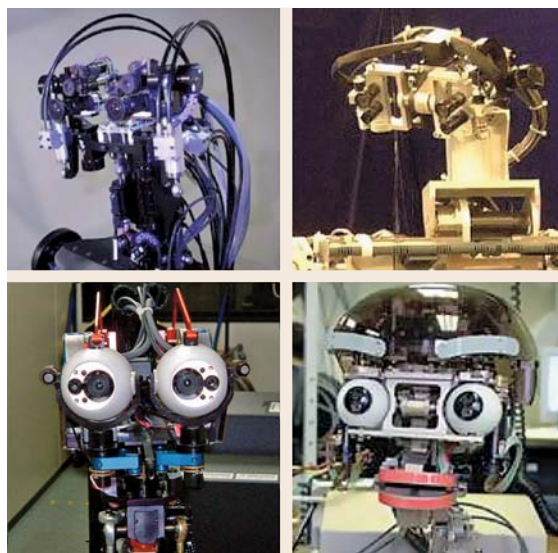


Fig. 56.39 The heads of humanoid robots come in many forms. A popular arrangement is to have two cameras per eye, as a crude approximation of foveal and peripheral vision in humans. *Top left*: biomimetic oculomotor control investigated on DB [56.98]. *Top right*: Cog's head [56.99]. *Bottom left*: the double-camera arrangement can be arranged in a less-double-barreled appearance (see Ude et al. [56.100] for more examples and an analysis) ©ATR; Humanoid head developed by ATR and SARCOS. *Bottom right*: the Infanoid robot [56.101]

each of the robot's eyes, but some researchers have also used special-purpose space-variant cameras modeled after the space-variant receptor densities in the human eye [56.102].

The eye movements of some humanoids are modeled explicitly after human eye movements. An example of a model of this kind is shown in Fig. 56.40. These bio-inspired approaches to active vision typically have four types of visual behavior:

Saccades. These are high-velocity movements to fixate a new target or *catch up* with a fast-moving target. From a control point of view, these movements are *ballistic* (at least in humans) – once initiated, they continue without responding to changing stimuli.

Smooth Pursuit. These are movements to continuously track a moving target. They apply at low velocities. These movements respond constantly to visual feedback about the target's location. A fast-moving target may also trigger small saccades.

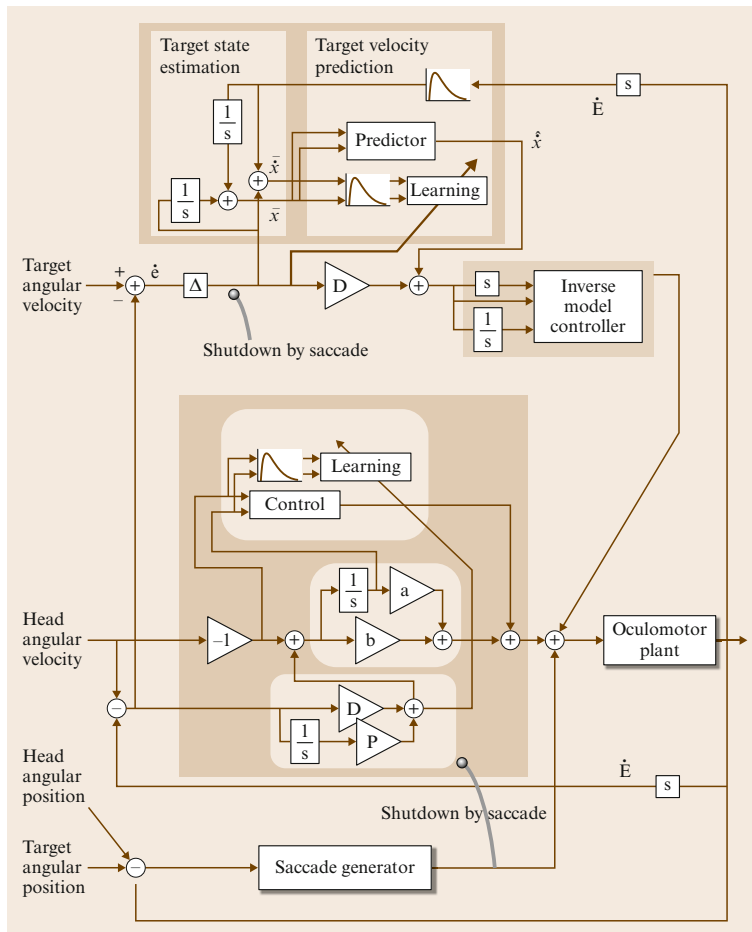


Fig. 56.40 A biomimetic control model [56.103], that integrates saccading, smooth pursuit, the vestibular-ocular reflex (VOR), and the optokinetic response (OKR). Smooth pursuit and VOR/OKR commands are summed, with periodic corrections to errors in position made by saccades

VOR and OKR. The vestibulo-ocular reflex and optokinetic response work to stabilize the direction of gaze in the presence of movement of the head and body, using inertial and visual information respectively.

Vergence. This movement drives the relative angle of the two eyes so that the same target is centered in both. This only applies to two-eyed systems that have this freedom of motion. For conventional stereo algorithms, vergence is a disadvantage, since the algorithms are simplest when the cameras remain parallel. Other algorithms are possible, but it is currently quite common not to use vergence.

56.6.2 Interpreting Human Expression

The interpretation of human expression is essential for many forms of natural human communication that could be valuable for humanoid robots.

Posture and Expression

The recognition and interpretation of the location and pose of humans is important, since humanoids are often expected to work in human environments. Algorithms for the following functions have been incorporated in various humanoids:

- Person finding
- Person identification
- Gesture recognition
- Face pose estimation

Asimo has used these functions to perform a prototypical reception task as shown in Fig. 56.41. The robot can find and identify a person, then recognize gestures such as *bye-bye*, *come here*, and *stop*, which are utilized for performing reception tasks. In general, such functions on a humanoid are not yet robust, and are active areas of research.



Fig. 56.41 Asimo recognizing a pointing gesture during a reception task

Speech Recognition

Speech is a natural, hands-free mode of communication between humans, and potentially between robots and humans. Speech recognition is a popular interface utilized for commanding a humanoid, and many off-the-shelf packages are now available. However the use of microphones embedded in the robot is problematic, because general-purpose speech recognition software is usually optimized for utterances captured by a microphone that is close to the speaker. In order to achieve sufficient recognition performance in natural interaction situations between a humanoid and a human new methods for speech recognition are being investigated. These methods compensate for sources of noise, such as the robot's motors and air flow in the environment, by using multiple microphones and multimodal cues [56.104]. However, at the time of writing researchers often circumvent these issues by using a headset, lavalier, or handheld microphones.

Auditory Scene Analysis

In order to attain more-sophisticated human–robot interaction, researchers have been developing methods for



Fig. 56.42 HRP-2 recognizing speech with background noises (TV sound)



Fig. 56.43 HRP-2 recognizing face and gaze direction for communication via eye contact

computational auditory scene analysis on a humanoid robot. The objective of this research is to understand an arbitrary sound mixture including nonspeech sounds and voiced speech, obtained by microphones embedded in the robot. Beyond speech recognition, this also involves sound-source separation and localization.

As for sound recognition, sound categories such as *coughing*, *laughing*, *beating by hand*, *adult's voice*, and *child's voice* have been shown to be recognizable using maximum-likelihood estimation with Gaussian mixture models. This function has been utilized during interactions between the HRP-2 and a human [56.105].

Multimodal Perception

Sound-source separation can be achieved by beam forming. In order to perform beamforming effectively,

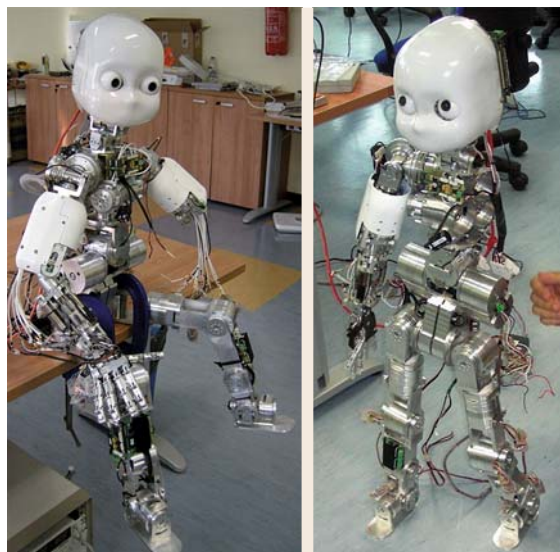


Fig. 56.44 The iCub robot [56.16], an infant-like robot that serves as an open humanoid platform for cognitive and neuroscience research

sound-source localization is essential. Vision can be utilized for finding the talker within the field of view. Hara et al. used a camera and an eight-channel microphone array embedded in the head of HRP-2, and succeeded in speech recognition in the presence of multiple sound sources by using sound source separation [56.106]. Figure 56.42 shows a scenario in which speech recognition is taking place with television (TV) sound playing in the background.

When integrated with speech recognition, vision can also help resolve the ambiguities of speech. For instance, the ambiguity of demonstrative pronouns such as *this* or *that* can sometimes be resolved by recognizing pointing gestures. Similarly, the face and gaze direction can be used to realize communication via eye contact, so that the humanoid only replies when a human is looking at it and talking to

it [56.105]. Multimodal interaction with these functions has also been demonstrated by HRP-2, as shown in Fig. 56.43.

56.6.3 Alternative Models for Human–Robot Communication

Humanoid robots typically use methods for perception and interaction that are similar to those found in fields such as computer vision and dialogue systems. There is also a subfield called *epigenetic* or *developmental* robotics that attempts to form an explicit bridge between robotics and human studies, such as psychology and neuroscience [56.107, 108]. Much of the research within this subfield deals with human-like perception and interaction using humanoid robots, such as the iCub shown in Fig. 56.44.

56.7 Conclusions and Further Reading

Because of the integrative nature of humanoid robotics, this chapter has avoided details and formalisms and liberally cross-referenced other chapters within the handbook that can provide the reader with deeper coverage of many of the areas of robotics on which humanoids depend. Additionally, this chapter thoroughly references work within the humanoid robotics community and related communities. Consequently, the reader should be able to use this chapter as a high-level overview of the field from which to explore specific areas of humanoid robotics.

Humanoid robotics is an enormous endeavor. The emulation of human-level abilities in a human-like robot serves as a grand challenge for robotics, with significant cultural ramifications. The motivations for humanoid robotics are as deep as they are diverse. From the earliest cave drawings, humanity has sought to represent itself. Robotics is one of the most recent mediums for this ongoing fascination. Besides this deep societal motivation, humanoid

robots offer unique opportunities for human–robot interaction, and integration into human-centric settings.

Over the last decade, the number of humanoid robots developed for research has grown dramatically, as has the research community. Humanoids have already gained a foothold in the marketplace as robots for entertainment through competitions and toys (e.g., Robo-One and RoboSapien). Time will tell whether consumers prefer robots with human-like qualities. For example, if service robots are to have two, high-degree-of-freedom arms and articulated sensing mechanisms, a human-like form may be a desirable configuration. Given the special properties of humanoid robots, they seem likely to at least fill a niche in the world of the future. Whether or not humanoids become a dominant form for robots may depend on the extent to which they can compete against specialized robots that are better suited to particular tasks, and human labor, which in some ways already meets the ultimate goals of humanoid robotics.

References

- 56.1 H. Inoue, S. Tachi, Y. Nakamura, K. Hirai, N. Ohyu, S. Hirai, K. Tanie, K. Yokoi, H. Hirukawa: Overview of humanoid robotics project of METI, Proc. 32nd Int. Symp. Robot. (ISR) (2001)
- 56.2 Y. Kuroki, M. Fujita, T. Ishida, K. Nagasaka, J. Yamaguchi: A small biped entertainment robot exploring attractive applications, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2003) pp. 471–476
- 56.3 C.G. Atkeson, J.G. Hale, F.E. Pollick, M. Riley, S. Kotosaka, S. Schaal, T. Shibata, G. Tevatia, A. Ude, S. Vijayakumar, M. Kawato: Using humanoid robots

- to study human behavior, *IEEE Intell. Syst.* **15**(4), 46–56 (2000)
- 56.4 K.F. MacDorman, H. Ishiguro: The uncanny advantage of using androids in social and cognitive science research, *Interact. Stud.* **7**(3), 297–337 (2006)
- 56.5 V. Bruce, A. Young: *In the Eye of the Beholder: The Science of Face Perception* (Oxford Univ. Press, Oxford 1998)
- 56.6 R. Blake, M. Shiffrar: Perception of human motion, *Annu. Rev. Psychol.* **58**, 47–73 (2007)
- 56.7 J.F. Werker, R.C. Tees: Influences on infant speech processing: Toward a new synthesis, *Annu. Rev. Psychol.* **50**, 509–535 (1999)
- 56.8 D. Hanson, A. Olney, I.A. Pereira, M. Zielke: Up-ending the uncanny valley, *Nat. Conf. Artif. Intell. (AAAI '05)*, Pittsburgh (2005)
- 56.9 K. Čapek's: *R.U.R. (Rossum's Universal Robots), A Play in Introductory Scene and Three Acts* (eBooks, Adelaide 2006), translated into English by D. Wyllie
- 56.10 *Metropolis*, directed by Fritz Lang (DVD) (Kino Video, 1927)
- 56.11 I. Asimov: *Caves of Steel* (Doubleday, Garden City, New York 1954)
- 56.12 B. Adams, C. Breazeal, R.A. Brooks, B. Scassellati: Humanoid robots: A new kind of tool, *IEEE Intell. Syst.* **15**(4), 25–31 (2000)
- 56.13 R.A. Brooks, L.A. Stein: Building brains for bodies, *Auton. Robot.* **1**(1), 7–25 (1994)
- 56.14 R.W. Gibbs Jr.: *Embodiment and Cognitive Science* (Cambridge Univ. Press, Cambridge 2006)
- 56.15 M. Lungarella, G. Metta: Beyond gazing, pointing, and reaching: A survey of developmental robotics, *Proc. Third Int. Workshop Epigenet. Robot.* (2003) pp. 81–89
- 56.16 G. Metta, G. Sandini, D. Vernon, D. Caldwell, N. Tsagarakis, R. Beira, J. Santos-Victor, A. Ijspeert, L. Righetti, G. Cappiello, G. Stellin, F. Becchi: The RobotCub project – an open framework for research in embodied cognition, *Proc. IEEE-RAS Int. Conf. Humanoid Robot.* (2005)
- 56.17 E. Grandjean, K. Kroemer: *Fitting the Task to the Human*, 5th edn. (Routledge, London 1997)
- 56.18 W. Karwowski: *International Encyclopedia of Ergonomics and Human Factors*, 2nd edn. (CRC, Boca Raton 2006)
- 56.19 R. Brooks, L. Aryananda, A. Edsinger, P. Fitzpatrick, C. Kemp, U.-M. O'Reilly, E. Torres-Jara, P. Varshavskaya, J. Weber: Sensing and manipulating built-for-human environments, *Int. J. Humanoid Robot.* **1**(1), 1–28 (2004)
- 56.20 W. Bluethmann, R. Ambrose, M. Diftler, S. Askew, E. Huber, M. Goza, F. Rehnmark, C. Lovchik, D. Magruder: Robonaut: A robot designed to work with humans in space, *Auton. Robot.* **14**(2–3), 179–197 (2003)
- 56.21 K. Yokoi, K. Nakashima, M. Kobayashi, H. Mihune, H. Hasunuma, Y. Yanagihara, T. Ueno, T. Gokyu, K. Endou: A tele-operated humanoid operator, *Int. J. Robot. Res.* **22**(5–6), 593–602 (2006)
- 56.22 S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, K. Ikeuchi: Generating whole body motions for a biped humanoid robot from captured human dances, *IEEE Int. Conf. Robot. Autom.* (2003) pp. 3905–3910
- 56.23 K. Yokoi, K. Kawauchi, N. Sawasaki, T. Nakajima, S. Nakamura, K. Sawada, T. Takeuchi, K. Nakashima, Y. Yanagihara, K. Yokohama, T. Isozumi, Y. Fukase, K. Kaneko, H. Inoue: Humanoid robot applications in HRP, *Int. J. Human. Robot.* **1**(3), 409–428 (2004)
- 56.24 B. Thibodeau, P. Deegan, R. Grunen: Static analysis of contact forces with a mobile manipulator, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA'06)* (2006) pp. 4007–4012
- 56.25 J. Gutman, M. Fukuchi, M. Fujita: Modular architecture for humanoid robot navigation, *Proc. 5th IEEE-RAS Int. Conf. Humanoid Robot.* (2005) pp. 26–31
- 56.26 K. Hirai, M. Hirose, Y. Haikawa, T. Takenaka: The development of Honda Humanoid Robot, *IEEE Int. Conf. Robot. Autom.* (1998) pp. 1321–1326
- 56.27 M.E. Rosheim: *Robot Evolution: The Development of Anthropotics* (Wiley, New York 1994)
- 56.28 M.E. Rosheim: *Leonardo's Lost Robots* (Springer, Berlin, Heidelberg 2006)
- 56.29 T.N. Hornyak: *Loving the Machine: The Art and Science of Japan's Robots* (Massachusetts Institute of Technology, Cambridge 2006)
- 56.30 J. Surrell: *Pirates of the Caribbean: From the Magic Kingdom to the Movies* (Disney, New York 2005)
- 56.31 J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, E. Thelen: Autonomous mental development by robots and animals, *Science* **291**(5504), 599–600 (2001)
- 56.32 J. Zlatev, C. Balkenius: Why "epigenetic robotics"? , *Proc. First Int. Workshop Epigenet. Robot. Model. Cognit. Dev. Robot. Syst.*, ed. by C. Balkenius, J. Zlatev, H. Kozima, K. Dautenhahn, C. Breazeal (Lund Univ. Press, Lund 2001) pp. 1–4
- 56.33 Y. Sodeyama, I. Mizuuchi, T. Yoshikai, Y. Nakanishi, M. Inaba: A shoulder structure of muscle-driven humanoid with shoulder blades, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'05)* (2005) pp. 4028–4033
- 56.34 I. Mizuuchi, M. Inaba, H. Inoue: A flexible spine human-form robot-development and control of the posture of the spine, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'01)*, Vol. 4 (2001) pp. 2099–2104
- 56.35 M. Vukobratović, J. Stepanenko: On the stability of anthropomorphic systems, *Math. Biosci.* **15**, 1–37 (1972)
- 56.36 S.H. Collins, A.L. Ruina, R. Tedrake, M. Wisse: Efficient bipedal robots based on passive-dynamic walkers, *Science* **307**, 1082–1085 (2005)

- 56.37 M.H. Raibert: *Legged Robots That Balance* (MIT Press, Cambridge 1986)
- 56.38 K. Nagasaka, K. Kuroki, S. Suzuki, Y. Itoh, J. Yamaguchi: Integrated motion control for walking, jumping and running on a small bipedal entertainment robot, *IEEE Int. Conf. Robot. Autom.* (2004) pp. 3189–3194
- 56.39 K. Fujiwara, F. Kanehiro, S. Kajita, K. Kaneko, K. Yokoi, H. Hirukawa: UKEMI: falling motion control to minimize damage to biped humanoid robot, *IEEE Int. Conf. Robot. Autom.* (2002) pp. 2521–2526
- 56.40 H. Hirukawa, S. Kajita, F. Kanehiro, K. Kaneko, T. Isozumi: The human-size humanoid robot that can walk, lie down and get up, *Int. J. Robot. Res.* **24**(9), 755–769 (2005)
- 56.41 Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, K. Fujimura: The intelligent ASIMO: system overview and integration, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'02)* (2002) pp. 2478–2483
- 56.42 K. Okada, S. Kagami, M. Inaba, H. Inoue: Plane segment finder: Algorithm implementation and applications, *Proc. Int. Conf. Robot. Autom. (ICRA'01)* (2001) pp. 2120–2125
- 56.43 K. Sabe, M. Fukuchi, J.-S. Gutmann, T. Ohashi, K. Kawamoto, T. Yoshigahara: Obstacle avoidance and path planning for humanoid robots using stereo vision, *Proc. Int. Conf. Robot. Autom. (ICRA'04)* (2004)
- 56.44 J. Gutman, M. Fukuchi, M. Fujita: Real-time path planning for humanoid robot navigation, *Proc. Int. Joint Conf. Artif. Intell.* (2005) pp. 1232–1238
- 56.45 T. Asfour, K. Regenstein, P. Azad, J. Schroder, A. Bierbaum, N. Vahrenkamp, R. Dillmann: ARMAR-III: An integrated humanoid platform for sensory-motor control, *Proc. 6th IEEE-RAS Int. Conf. Humanoid Robot.* (2006)
- 56.46 C. Ott, O. Eiberger, W. Friedl, B. Bäuml, U. Hillenbrand, C. Borst, A. Albu-Schäffer, B. Brunner, H. Hirschmüller, S. Kielhöfer, R. Konietzschke, M. Suppa, T. Wimböck, F. Zacharias, G. Hirzinger: A humanoid two-arm system for dexterous manipulation, *Proc. IEEE-RAS Int. Conf. Humanoid Robot.* (2006) pp. 276–283
- 56.47 M. Vande Weghe, M. Rogers, M. Weissert, Y. Matsuoka: The ACT hand: Design of the skeletal structure, *Proc. 2004 IEEE Int. Conf. Robot. Autom. (ICRA '04)* (2004)
- 56.48 L.Y. Chang, Y. Matsuoka: A kinematic thumb model for the ACT hand, *Proc. IEEE Int. Conf. Robot. Autom. (ICRA '06)* (2006) pp. 1000–1005
- 56.49 C. Lovchik, M.A. Diftler: The robonaut hand: A dexterous robot hand for space, *Proc. Int. Conf. Robot. Autom. (ICRA)* (1999) pp. 907–912
- 56.50 M.J. Marjanović: Teaching an Old Robot New Tricks: Learning Novel Tasks via Interaction with People and Things. Ph.D. Thesis (MIT, 2003)
- 56.51 L. Natale, E. Torres-Jara: A sensitive approach to grasping, *Sixth Int. Conf. Epigenet. Robot., Paris* (2006) pp. 20–22
- 56.52 A. Edsinger, C.C. Kemp: Manipulation in human environments, *Proc. IEEE/RAS Int. Conf. Humanoid Robot., Genoa* (2006)
- 56.53 M. Santello, M. Flanders, J.F. Soechting: Postural hand synergies for tool use, *J. Neurosci.* **18**(23), 10105–10115 (1998)
- 56.54 F. Tomita, T. Yoshimi, T. Ueshiba, Y. Kawai, Y. Sumi, T. Matsushita, N. Ichimura, K. Sugimoto, Y. Ishiyama: R&D of versatile 3D vision system VVV, *Proc. IEEE Int. Conf. SMC, Vol. 5* (1998) pp. 4510–4516
- 56.55 K. Yokoi, N.E. Sian, T. Sakaguchi, H. Arisumi, E. Yoshida, Y. Kawai, K. Maruyama, T. Yoshimi, O. Stasse, S. Kajita: Humanoid Robot HRP-2 No.10 with human supervision, *Proc. Int. Symp. Robot. (ISR'05)* (2005)
- 56.56 K. Yokoyama, H. Handa, T. Isozumi, Y. Fukase, K. Kaneko, F. Kanehiro, Y. Kawai, F. Tomita, H. Hirukawa: Cooperative works by a human and humanoid robot, *Proc. Int. Conf. Robot. Autom. (ICRA'03)* (2003) pp. 2985–2991
- 56.57 E. Huber, K. Baker: Using a hybrid of silhouette and range templates for real-time pose estimation, *Proc. ICRA 2004 IEEE Int. Conf. Robot. Autom., Vol. 2* (2004) pp. 1652–1657
- 56.58 A. Edsinger, C.C. Kemp: Two arms are better than one: A behavior-based control system for assistive bimanual manipulation, *Proc. 13th Int. Conf. Adv. Robot. (ICAR)* (2007)
- 56.59 P. Fitzpatrick, G. Metta: Grounding vision through experimental manipulation, *Phil. Trans. R. Soc. Math. Phys. Eng. Sci.* **361**(1811), 2165–2185 (2003)
- 56.60 V. Zatsiorsky: *Kinetics of Human Motion* (Human Kinetics, Champaign 2002)
- 56.61 M. Darainy, F. Towhidkhan, D.J. Ostry: Control of hand impedance under static conditions and during reaching movement, *J. Neurophysiol.* **97**, 2676–2685 (2007)
- 56.62 S. Schaal, D. Sternad, R. Osu, M. Kawato: Rhythmic arm movement is not discrete, *Nat. Neurosci.* **7**, 1136–1143 (2004)
- 56.63 M. Williamson: Robot Arm Control Exploiting Natural Dynamics. Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1999)
- 56.64 G. Pratt, M. Williamson: Series elastic actuators, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS-95)*, Pittsburg, Vol. 1 (1995) pp. 399–406
- 56.65 A. Ijspeert, J. Nakanishi, S. Shaal: Learning rhythmic movements by demonstration using nonlinear oscillators, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS2002)* (2002) pp. 958–963
- 56.66 J. Glassmire, M. O'Malley, W. Bluethmann, R. Ambrose: cooperative manipulation between humans and teleoperated agents, *Haptics* **00**, 114–120 (2004)

- 56.67 A. Edsinger, C.C. Kemp: Human-robot interaction for cooperative manipulation: Handing objects to one another, Proc. 16th IEEE Int. Symp. Robot Human Interact. Commun. (RO-MAN) (2007)
- 56.68 S. Schaal: Is imitation learning the route to humanoid robots?, Trends Cognit. Sci. **3**(6), 233–242 (1999)
- 56.69 A. Billard, M.J. Mataric: A biologically inspired robotic model for learning by imitation, Proc. Fourth Int. Conf. Auton. Agents, ed. by C. Sierra, M. Gini, J.S. Rosenschein (ACM, Barcelona 2000) pp.373–380
- 56.70 R. Zollner, T. Asfour, R. Dillmann: Programming by demonstration: Dual-arm manipulation tasks for humanoid robots, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS) (2004)
- 56.71 D.C. Bentivegna, A. Ude, C.G. Atkeson, G. Cheng: Learning to act from observation and practice, Int. J. Humanoid Robot. **1**(4), 585–611 (2004)
- 56.72 R. Platt, A. Fagg, R. Grunert: Re-using schematic grasping policies, Proc. IEEE-RAS Int. Conf. Humanoid Robot. (2005)
- 56.73 K. Harada, S. Kajita, H. Saito, M. Morisawa, F. Kanehiro, K. Fujiwara, K. Kaneko, H. Hirukawa: A humanoid robot carrying a heavy object, Proc. IEEE Int. Conf. Robot. Autom. (2005) pp.1724–1729
- 56.74 K. Yamane: *Simulating and Generating Motions of Human Figures* (Springer, Berlin, Heidelberg 2004)
- 56.75 E.S. Neo, K. Yokoi, S. Kajita, K. Tanie: A framework for remote execution of whole body motions for humanoid robots, IEEE-RAS Int. Conf. Humanoids (2004) pp. 58–68
- 56.76 E.S. Neo, K. Yokoi, S. Kajita, F. Kanehiro, K. Tanie: A switching command-based whole-body operation method for humanoid robots, IEEE/ASME Trans. Mechatron. **10**(5), 2569–2574 (2005)
- 56.77 J.J. Kuffner, K. Nishiwaki, S. Kagami, M. Inaba, H. Inoue: Motion planning for humanoid robots under obstacle and dynamic balance constraints, IEEE Int. Conf. Robot. Autom. (2001) pp.692–698
- 56.78 K. Yamane, Y. Nakamura: Dynamics filter-concept and implementation of online motion generation for human figures, IEEE Trans. Robot. Autom. **19**(3), 421–432 (2003)
- 56.79 J.J. Kuffner, S.M. LaValle: RRT-Connect: An efficient approach to single-query path planning, IEEE Int. Conf. Robot. Autom. (2000) pp.995–1001
- 56.80 J. Yamaguchi, E. Soga, S. Inoue, A. Takanishi: Development of a bipedal humanoid robot – control method of whole body cooperative dynamic biped walking –, IEEE Int. Conf. Robot. Autom. (1999) pp.369–374
- 56.81 S. Kagami, F. Kanehiro: AutoBalancer: An on-line dynamic balance compensation scheme for humanoid robots, Workshop Algorithmic Found. Robot. (2000) pp.79–89
- 56.82 O. Khatib: A unified approach to motion and force control of robot manipulators: The operational space formulation, Int. J. Robot. Autom. **3**(1), 43–53 (1987)
- 56.83 L. Sentis, O. Khatib: A whole-body control framework for humanoids operation in human environments, IEEE Int. Conf. Robot. Autom. (2006) pp.2641–2648
- 56.84 J. Park, O. Khatib: Contact consistent control framework for humanoid robots, IEEE Int. Conf. Robot. Autom. (2006) pp.1963–1969
- 56.85 S. Kajita, F. Kanehiro, K. Fujiwara, K. Harada, K. Yokoi, H. Hirukawa: Resolved momentum control: Humanoid motion planning based on the linear and angular momentum, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003) pp.1644–1650
- 56.86 K. Kaneko, F. Kanehiro, S. Kajita, H. Hirukawa: Humanoid Robot HRP-2, IEEE Int. Conf. Robot. Autom. (2004) pp.1083–1090
- 56.87 L. Sentis, O. Khatib: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives, Int. J. Humanoid Robot. **2**(4), 505–518 (2005)
- 56.88 K. Harada, S. Kajita, F. Kanehiro, K. Fujiwara, K. Kaneko, K. Yokoi, H. Hirukawa: Real-time planning of humanoid robot's gait for force controlled manipulation, Proc. IEEE Int. Conf. Robot. Autom. (2004) pp. 616–622
- 56.89 N. Mansard, O. Stasse, F. Chaumette, K. Yokoi: Visually-Guided Grasping while Walking on a Humanoid Robot, IEEE Int. Conf. Robot. Autom. (ICRA'07) (2007)
- 56.90 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: Pushing manipulation by humanoid considering two-kinds of ZMPs, Proc. IEEE Int. Conf. Robot. Autom. (2003) pp.1627–1632
- 56.91 K. Harada, S. Kajita, K. Kaneko, H. Hirukawa: Dynamics and balance of a humanoid robot during manipulation tasks, IEEE Trans. Robot. **22**-3, 568–575 (2006)
- 56.92 H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, M. Morisawa: A universal stability criterion of the foot contact of legged robots – Adios ZMP, Proc. IEEE Int. Conf. Robot. Autom. (2006) pp.1976–1983
- 56.93 Y. Hwang, A. Konno, M. Uchiyama: Whole body co-operative tasks and static stability evaluations for a humanoid robot, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2003) pp.1901–1906
- 56.94 T. Takenaka: Posture Control for a Legged Mobile Robot, Japanese Patent 10230485 (1998)
- 56.95 K. Inoue, H. Yoshida, T. Arai, Y. Mae: Mobile manipulation of humanoids – real-time control based on manipulability and stability –, Proc. IEEE Int. Conf. Robot. Autom. (2000) pp.2217–2222
- 56.96 T. Takubo, K. Inoue, K. Sakata, Y. Mae, T. Arai: Mobile manipulation of humanoid robots – control method for CoM position with external force –,

- Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2004) pp.1180–1185
- 56.97 C. Breazeal, A. Edsinger, P. Fitzpatrick, B. Scassellati, P. Varchavskaya: Social constraints on animate vision, *IEEE Intell. Syst.* **15**, 32–37 (2000)
- 56.98 T. Shibata, S. Vijayakumar, J. Conradt, S. Schaal: Biomimetic oculomotor control, *Adapt. Behav.* **9**, 189–208 (2001)
- 56.99 B. Scassellati: *A Binocular, Foveated Active Vision System*, Vol. AIM-1628 (MIT, Cambridge 1998)
- 56.100 A. Ude, C. Gaskett, G. Cheng: Foveated vision systems with two cameras per eye, *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando (2006)
- 56.101 H. Kozima: Infanoid: A babybot that explores the social environment. In: *Socially Intelligent Agents: Creating Relationships with Computers and Robots*, ed. by K. Dautenhahn, A.H. Bond, L. Canamero, B. Edmonds (Kluwer Academic, Amsterdam 2002) pp.157–164
- 56.102 F. Berton, G. Sandini, G. Metta: Anthropomorphic Visual Sensors. In: *The Encyclopedia of Sensors*, Vol.X, ed. by M.V. Pishko, C.A. Grimes, E.C. Dickey (American Scientific, Valencia 2005) pp.1–16
- 56.103 T. Shibata, S. Schaal: Biomimetic gaze stabilization based on feedback-error-learning with nonparametric regression networks, *Neural Netw.* **14**(2), 201–216 (2001)
- 56.104 P. Heracleous, S. Nakamura, K. Shikano: Simultaneous recognition of distant-talking speech to multiple talkers based on the 3-D N-best search method, *J. VLSI Signal Process. Syst. Archive* **36**(2–3), 105–116 (2004)
- 56.105 J. Ido, Y. Matsumoto, T. Ogasawara, R. Nisimura: Humanoid with interaction ability using vision and speech information, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'06)* (2006)
- 56.106 I. Hara, F. Asano, H. Asoh, J. Ogata, N. Ichimura, Y. Kawai, F. Kanehiro, H. Hirukawa, K. Yamamoto: Robust speech interface based on audio and video information fusion for humanoid HRP-2, *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS'04)* (2004) pp.2402–2410
- 56.107 M. Lungarella, G. Metta, R. Pfeifer, G. Sandini: Developmental robotics: a survey, *Connect. Sci.* **15**(4), 151–190 (2003)
- 56.108 B. Scassellati, C. Crick, K. Gold, E. Kim, F. Shic, G. Sun: Social development, *Comput. Intell. Mag. (IEEE)* **1**(3), 41–47 (2006)

57. Safety for Physical Human–Robot Interaction

Antonio Bicchi, Michael A. Peshkin, J. Edward Colgate

In this chapter, we report on different approaches to dealing with the problem of achieving the best performance under the condition that safety is provided throughout task execution. We also report on intelligent assist devices (IADs) that go beyond conventional notions of robot safety to protect human operators from harm, such as cumulative trauma disorders. However, IADs and other physical human–robot interaction (pHRI) devices are themselves generally powerful enough to cause harm. We argue that the differences between pHRI applications and conventional industrial manipulation requires that safety and reliability standards be rethought, and we offer a preview of the directions currently being undertaken by international standards committees.

We chapter discuss the new frontiers of robotic physical interaction with humans, describing

57.1 Motivations for Safe pHRI	1336
57.2 Safety for Hands-Off pHRI	1337
57.3 Design of Intrinsically Safe Robots	1338
57.4 Safety for Hands-On pHRI.....	1341
57.5 Safety Standards for pHRI.....	1345
57.6 Conclusions	1346
References	1346

motivations and applications of safe pHRI. The state of the art, and the technical challenges to develop new robotic systems for safe and effective collaboration with people, are discussed, subdividing the exposition into hands-off and hands-on pHRI systems. We present an overview of the applicable safety standards and their ongoing development.

Popular notions of robotics have long foreseen humans and robots existing side by side, sharing work, or even (as cyborgs) integrating into a greater whole. Until very recently, the reality has been quite different. Industrial robots have been far too dangerous to share space with humans, while physical interaction has (except in the case of telemanipulators, which date back to the 1940s) been far too unintuitive.

Shortly, all this is going to change radically: the next generation of robots will interact with people directly. Human–robot interaction (HRI) will certainly happen at the cognitive level (cHRI, Chap. 58), fundamentally concerning communication between human and robot through the many channels available to us (video displays, sounds, watching each other’s motions, spoken language, or even gaze direction or facial expression). However, robots are distinct from computers in that they physically embody the link between perception and

action. Hence, one of the most revolutionary and challenging features of the next generation of robots will be physical human–robot interaction (pHRI). In pHRI, humans and robots share the same workspace and come into contact with each other. Physical interaction may happen occasionally if normal operation is intended to be without contact, or on purpose if the operator is supposed to work in physical contact with the machine, exchanging forces and cooperating in action upon on the environment. We will designate these intended forms of interaction *hands-off* and *hands-on* pHRI, respectively.

pHRI robots will be designed to coexist and cooperate with humans in applications such as assisted industrial manipulation, collaborative assembly, domestic work, entertainment, rehabilitation or medical applications. Clearly, such robots must fulfill different requirements from those typically met in conventional industrial applications: while it might be possible to re-

lax requirements on velocity of execution and absolute accuracy, concerns such as safety and dependability become of paramount importance when human lives are

involved. This paradigm shift entails many differences in design and control, and opens up new challenging directions for research.

57.1 Motivations for Safe pHRI

Since the very beginning of industrial robotics, a great deal of attention has been paid to robot safety. The first line of defense has always been to take all measures to enforce segregation between robots and people. It is on such philosophy that most of the early technical literature on the topic of robotic safety is based [57.2–4], as well as the existing robotics safety regulations and standards [57.5–7]. However, the segregation paradigm fails in cases where the human and the robot must share the physical environment and in applications in which successful task completion requires collaboration.

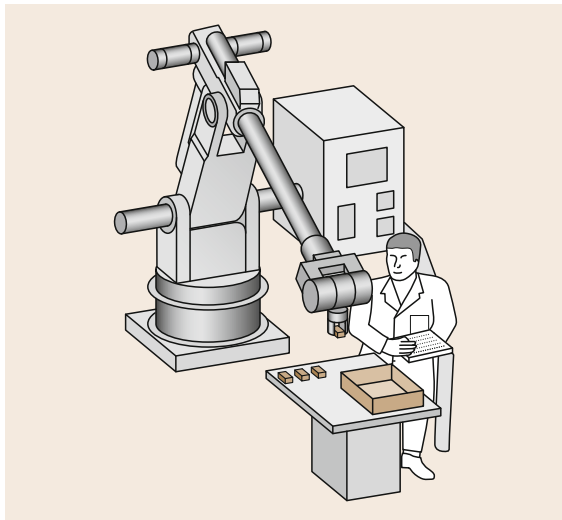


Fig. 57.1 Hazard in lead-through teaching [57.1]

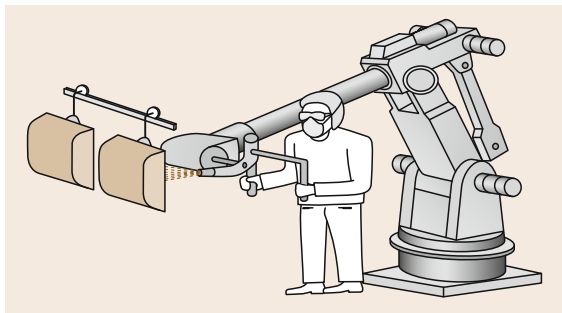


Fig. 57.2 pHRI in walk-through teaching [57.1]

Data on industrial robot-related fatalities indicate that, even in traditional applications of industrial robots, safety is not a solved problem – especially because of all the operational phases where the human operator is by necessity physically close to the mechanical arm or vehicle.

Studies reported in [57.1] show that *many robot accidents do not occur under normal operating conditions but instead during programming, program touch-up or refinement, maintenance, repair, testing, setup, or adjustment. During many of these operations the operator, programmer, or corrective maintenance worker may temporarily be within the robot's working envelope where unintended operation could result in injuries.*

People in the manufacturing environment are often injured even in the absence of robots, and curiously, this provides a major motivation for hands-on pHRI. Repetitive manual material handling exposes people to work-related musculoskeletal disorders (WMSD). In 1990, the average US worker's compensation expense for a lost-time back injury was \$24 000 (source: National Council of Compensation Insurance). The ergonomics and productivity consequences are also documented for US industries: in 2004, over 400 000 WMSD were reported involving days away from work. Approximately one-third of these injuries were in the goods-producing industries, including manufacturing [57.8]. The total cost of these and related problems is of the order of \$13–20 billion annually in the US alone. In many cases, robotic assistance could reduce WMSD drastically.

Market pressures together with ethical concerns are about to topple some of the barriers separating robots and people, and thus require a next generation of machines for safer pHRI. Consider for example the automotive industry. While the advent of robots has mitigated harsh, unsafe conditions in the body shop (where sheet metal is welded into a structure), and the paint shop, other areas have gone untouched for over three decades. The general assembly area, where the engine and cockpit subsystems and seats and tires are integrated with the painted shell, is such an untouched area. The tooling that has been used tends to be mechanical in nature and is primarily powered by human and pneumatic effort. Sensing and decision-making are the worker's responsibility. The

principal reasons for not automating general assembly are both technical and economic. From a technological perspective, using robots for assembly in processes with high geometric dimensional variability is yet to be achieved with the reliability levels required for high-volume production. Further, programming complexity grows greatly with the number of trim options offered to the customer (e.g., leather seats, two-tone color, V6 engine, overhead console). Financially, the necessary increase in physical floor space dramatically impacts costs. In summary, the worker – with unsurpassed sensing and processing abilities – is a critical component in the assembly process. The primary concern, then, is that of the worker's well being, given that he/she tends to tire and is susceptible to injuries resulting from cognitive LAPSES and motor effort.

The case of general assembly explains why robots sometimes fail to replace people, but also motivates the close cooperation of robots and people. Humans have dense sensors (human tactile sensing especially outstripping attempts at emulation) and humans have an ability to interpret sensory data which artificial intelligence cannot approach. However humans have poor open-loop accuracy, tire easily, and are subject to repetitive stress injuries. In contrast robots have minimal sensing, not for lack of sensors but for their inability to interpret sensory input. However robots have high accuracy and speed

and work indefinitely. Even with increasing sensorization, robots are not about to match a human's literally millions of sensory receptors.

The characteristics of robots and humans have led to a new generation of safe human-interacting machines known as intelligent assist devices (IADs). IADs are a class of robots intended principally for *comanipulation* of payloads along with a human partner. IADs serve principally to augment the strength of a human, but they may also serve to guide motion via virtual surfaces, tracking of a moving assembly line, or the like. Research and development into IADs is actively stimulated by automotive industries (General Motors and Ford Motor in the US, and Toyota in Japan taking the lead in the development of technologies and new standards). The value of robot–human collaboration is also being discovered in a variety of nonindustrial environments: from exoskeletons as human power amplifiers [57.9] or as haptic interfaces in virtual-reality environments [57.10], to medical assistants and telesurgery [57.11] in an operating room, to rehabilitation [57.12, 13] and robotic sports-trainers [57.14].

In the rest of this chapter, we will consider safety in two distinct cases of human–robot interaction: first, when physical contact occurs occasionally or by accident, and second, when physical interaction is an intentional part of planned operations.

57.2 Safety for Hands-Off pHRI

Management of risk for humans working near robots involves in general very broad considerations, ranging from potential electrical and pressurized fluid hazards, pinching hands or feet, dropping parts, etc. These however fall beyond the scope of this chapter, which focuses on the most dangerous risk specific to robots: the situation in which, in an unspecified instant during execution of a robot movement, a collision or unwanted force exertion occurs between the machine and a human.

Even under this restriction, the safety of pHRI involves several different aspects and depends on many factors, ranging from software dependability, to possible mechanical failures, to human errors in interfacing with the machine, etc. A thorough hazard analysis and risk evaluation must be performed according to specifically designed procedures: these methods are receiving a growing attention from the research community [57.15–18], international standardization bodies (Sect. 57.3), and industries. As an example, KUKA's *Safe Operation* and *Safe Handling* systems, incorporat-

ing Pilz's safety-related fieldbus, is currently in use at BMW's *Body in White* lines [57.19], and allows safe and redundant monitoring of the robot's position, velocity, and acceleration. Such a technology provides the basis for safe human–robot cooperation and can be used, e.g., for manual robot guidance with safely reduced speed. In the case of a malfunction (hardware or software) the robot is stopped immediately. Whether immediate stops are sufficient to guarantee safety, or even are the safest action to undertake, is not certain.

The *holy grail* of pHRI design is *intrinsic safety*: that is, a robot that will be safe to humans no matter what failure, malfunctioning, or even misuse might happen. Naturally, perfect safety against all odds is not feasible for machines which have to deliver performance in terms of weight lifting, swift motion, etc.: the trade-off between safety and performance is inherent to pHRI.

One important aspect of designing robots for safety is the ability to quantitatively assess the risk of injuries in accidents, so as to be able to compare different design

and control solutions, and optimize them. We will focus here only on a particular aspect of safety of robot manipulators, that of unexpected collisions by the manipulator with a human operator, which in the worst case could happen anywhere on the manipulator structure and on the body of the operator and at any time during the execution of a planned trajectory. Much can be learned from the biomechanics of impacts in sports and automotive applications. In the relevant literature [57.20–22], several indices of the severity of an impact are proposed, which can be mapped (through extensive experimental campaigns and statistical correlations) to the probability of causing a certain level of injury. Among these are, e.g., Gadd's severity index (GSI), the 3 ms criterion, the viscous injury response (VC), and the thoracic trauma index (TTI). The most widely used index in the automotive industry is presently the so-called head injury criterion (HIC). Most of these indices are related to a basic *tolerance curve* developed at Wayne State University (the so-called WSTC) on the basis of data experimentally acquired from animal and cadaver head collision tests. The WSTC is a curve plotting head accelerations versus impact duration, indicating that very intense head acceleration is tolerable if it is very brief, but that much less is tolerable if the pulse duration exceeds 10–15 ms (as the time of exposure to cranial pressure pulses increases, the tolerable intensity decreases).

Gadd [57.20] plotted the tolerance limit curve in log–log coordinates, obtaining a straight line of slope ≈ 2.5 , and proposed accordingly a severity index as

$$\text{GSI} = \int_0^t a^{2.5} d\tau,$$

where a is the head acceleration (in g), and the integral is extended to the whole duration of collision. The head

injury criterion (HIC) was proposed by Versace [57.22] as a refinement of the GSI, defined as

$$\text{HIC} = T \left[\frac{1}{T} \int_0^T a(\tau) d\tau \right]^{2.5},$$

where T is conventionally the *duration* of the impact. As the choice of this duration is often difficult, it is recommended to consider the worst-case HIC at varying T . In general, both ends of the interval $T = (t_2 - t_1)$ are varied, and time t_2 is close to the time at which the head reaches its maximum velocity (typically, $t_2 - t_1 \leq 15$ ms). Generalizations of the HIC to collisions with other parts of the body have been proposed whereby the 2.5 exponent is replaced by other empirically determined values.

These metrics have constituted a useful basis for starting the development and evaluation of safe robotic concepts [57.23, 24]. The conditions under which these indices were formulated are quite different from those actually encountered in robotics. A recent experimental campaign [57.25] to measure the effects of impacts of a robot arm using standard crash-test facilities indicated that classical severity indices established in the automobile industry cannot be transferred without correction to the field of robotics. Indeed, the cited paper extrapolates experimental and simulation data to suggest that any robot, no matter how massive, whose parts are moving with a velocity of up to 2 m/s, would not cause impacts with a HIC larger than ≈ 100 , which is similar to what would occur to a person quickly walking and hitting a wall head first. Although such impacts, according to accepted standards in crash tests [57.26], would be considered to provoke only *very low injury potential*, they are still clearly unacceptable risks for a robot intended to coexist safely with humans.

57.3 Design of Intrinsically Safe Robots

Researchers have attempted to design intrinsically safer robots in various ways. The inherent danger of conventional robot arms might be mitigated by increasing their sensorial apparatus, using, e.g., proximity-sensitive skins such as those proposed in [57.27, 28], or by increasing the energy-absorbing properties of protective layers (adding enough soft and compliant coverings, or placing airbags around the arm, etc.).

An approach that has been intensively explored since the early 1980s is to design active force controllers,

e.g., Salisbury's stiffness control or Hogan's impedance control schemes, to introduce compliance with respect to sensed interactions. However, these approaches may not prove robust with respect to impacts on portions of the arm that are not equipped with force/torque sensors. Also, it is well known in the robotics literature that there are intrinsic limitations to what the controller can do to alter the behavior of the arm if the mechanical bandwidth (basically dictated by mechanism inertia and friction) is not matched to the task [57.29]. In other



Fig. 57.3 The **WAM** robot (Barrett Technology Inc., Cambridge)

words, making a rigid, heavy robot behave gently and safely is an almost hopeless task, if realistic conditions are taken into account. Probably, the first arm with a lightweight structure intended for service applications was the whole-arm manipulator (**WAM**) proposed in [57.30].

The focus on safety has in recent years encouraged a number of technological innovations in actuators, sensors, and structural design, leading to impressive realizations such as the several generations of Deutsches



Fig. 57.4 The third version of the **DLR** lightweight robot (**LWR-III**) is capable of operating a payload equal to its own weight (13.5 kg)

Zentrum für Luft- und Raumfahrt (**DLR**) lightweight robots [57.31].

Arms in this class are primarily characterized by a low inertia of their moving parts and back-drivability. The **DLR LWRIII** arm uses high-performance actuators and integrated torque sensors, which allow sophisticated control algorithms to be run. Significant joint compliance is often present in these arms as a side-effect of other design choices such as use of cable transmissions, harmonic drives, or joint torque sensors. Suitable force-control policies have been designed to employ such arms in safety-critical applications [57.32–35].

Another approach to increasing the safety level (and psychological acceptability) of robot arms interacting with humans is to intentionally introduce mechanical compliance into the design. By this measure, which complements low-inertia design of the arm’s links and soft coverings, researchers can dynamically decouple the actuators’ rotor inertia from the links whenever an impact occurs. Indeed, motors contribute most of the effective inertia in conventional geared drives, while direct-drive solutions remain impractical due to their large size and cost. Naturally, compliant transmissions can diminish performance, causing slow response, increased oscillations and longer settling times. However, this might not be a problem in some applications, e.g., entertainment. In most practical applications other than entertainment, however, positioning accuracy and velocity of task execution are crucial: controlling *soft* manipulators quickly and accurately – similarly to what humans do with their own arms – is one of the main avenues open in front of **pHRI** research.

The problem of controlling passively elastic joints so as to recover performance has been studied in depth in the robotics literature, both in the general case (see, e.g., [57.36, 37] and the review in [57.38]) and in safety-



Fig. 57.5 An entertainment robot by **SARCOS** physically interacting with a bystander. Compliance is built into the actuation system; accuracy and velocity are not primary goals

oriented design contexts [57.39, 40]. Adaptive control methods that could cope with uncertain inertia in robots with elastic joints have also been studied [57.41], although adapting to large, uncertain stiffness parameters appears to be an open problem.

It can be expected that the promptness of an elastically actuated arm is intrinsically severely reduced, if compliance is high enough to be effective for safety. In other words, even if optimal methods for controlling very compliant arms were available, there are inherent limitations on performance imposed by such hardware. To overcome these limitations, a recent trend in intrinsically safe robotics advocates the co-design of the mechanics and control of passively compliant, yet fast, strong, and accurate arms.

Very compliant transmissions may ensure natural and safe interaction but be inefficient in transferring energy from actuators to the links for fast motion. An approach to reducing the inertia of a manipulator arm for safety, while preserving performance, is distributed macro-mini actuation (DM2) [57.23, 42]. For each degree of freedom (joint), a pair of actuators is employed, connected in parallel and located in different parts of the manipulator. The key of the macro-mini actuation approach is to divide the torque generation into separate low- and high-frequency components which sum in parallel, according to an idea originally proposed in [57.43]. Gravity compensation and other large but slowly time-varying torques are generated by a *series elastic actuator* (SEA) [57.39] consisting of a relatively large actuator located at the base of the manipulator and connected to the axis through a spring, thus achieving low overall impedance. For the high-frequency torques needed, small motors collocated at the joints are used, providing high-performance motion while not significantly increasing the combined impedance of the manipulator-actuator system.

A different approach to gain performance for guaranteed-safety joint actuation is to allow the mechanical compliance (or, more generally, the impedance) of motion transmission devices to vary during the execution of tasks. As often, an inspiration can be taken from nature, with the human arm representing an instance of a powerful, agile manipulator that is at the same time safe and capable of high delicacy. The example from nature suggests that *variable compliance* can be extremely useful towards the goal of safe and fast motion: humans can vary the compliance of their arm for different tasks, and even during different phases of a task [57.44]. Unless it is intentionally used to harm, the human arm is typically controlled to move slowly when it is stiff, and

to be compliant while moving fast. This behavior also shows how the safety-performance trade-off can be addressed not by mere mechanical means, nor by control strategies alone, but by the intimate combination of the two aspects.

The idea of adapting the mechanical compliance of a robot arm to the task is almost as old as robotics itself ([57.45]) and has seen numerous implementations in different guises. Among the designs oriented towards physical human-robot interaction, we mention the mechanical impedance adjuster (MIA) [57.46] adopted in the humanoid robot WENDY, which uses a variable-length leaf spring. The MIA can realize a wide range of compliance values to adapt to different tasks in pHRI. However, similarly to many other such devices, the design mainly aims at adapting the compliance to a given task before the execution of the task itself, and keeps it constant during motion.

A different approach, called variable-impedance actuation (VIA) [57.24, 47] proposes a mechanical/control co-design that exploits rapid and continuous variations of transmission impedance during task execution (by varying its stiffness, damping, or gear ratio), so as to provide low levels of injury risk at any time during motion and minimize negative effects on control performance.

The trade-off between safety and performance also involves task motion planning. For a mechanism with given total inertia and actuator torque limits, one can formulate an optimal control problem to be used for comparing mechanical/actuation alternatives at their best possible performance. This question can be formalized as a minimum-time optimal control problem (i. e., a problem in the class of the so-called *brachistochrones*), specified for given initial and final positions of the joints and of the motors, and has to be computed taking into ac-

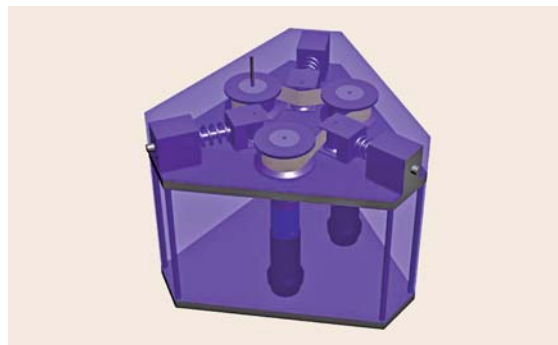


Fig. 57.6 The variable stiffness actuator of the University of Pisa implements the VIA approach by an antagonistic arrangement of nonlinear springs

count the limits on maximum motor torque and the lower and upper bounds to available transmission impedance. Admissible trajectories must satisfy a safety condition, which is expressed as a function of the velocities of the manipulator, its configuration, and the amount of inertia that is reflected on the links through the compliant transmission.

Solutions to such a *safe brachistochrone* problem for a single-joint actuator-link **VIA** system were obtained by numerical optimization in [57.24], showing that the optimal control policy consists of imposing high stiffness at low speeds (when the link needs accuracy and control authority to accelerate promptly), while low stiffness should be used at high speeds, to reduce the effect of the reflected motor inertia in the initial instants of impact. Such a *stiff and slow/soft and fast* control paradigm for **VIA** systems matches well with physical intuition (most of the motion energy transfer from the motor should occur during the initial and final acceleration/deceleration phases), and with the natural control inspiration. Although no solutions are known to the multilink safe brachistochrone problem to date, ideal solutions ob-

tained for a single joint can provide guidelines to be used also for more complex systems. The **VIA** concept can be realized though different arrangements, including the relocation of actuators close to the robot base and transmission of motion through steel cables and pulleys, the use of parallel and distributed macro–mini actuation with elastic couplings, the use of antagonistic actuation schemes, etc.

Once a collision has occurred, and its primary impact effects have been possibly reduced by mechanical design, further injuries can be avoided by a suitable reflex control action. A collision detection scheme based on the command inputs and sensor outputs available on the robot should trigger the switching of the control law [57.48, 49]. A physical collision can be treated as a fault condition of the robotic device. Accordingly, methods and algorithms available for the so-called fault detection and isolation problem [57.50] have recently been applied to robot collision detection [57.51, 52]. Their integration into a safety-oriented controller and their extension to the case of (possibly time-varying) compliant robot arms is an open issue for research.

57.4 Safety for Hands-On pHRI

A different set of safety methods and strategies are required in cases where successful task completion requires people and robots to collaborate intimately. This is the case for intelligent assist devices (**IADs**), human extenders, and collaborative robots (cobots).

IADs have their roots in *Ralph Mosher's* famous Handiman and Hardiman projects of the 1960s [57.54]. It was the initiative of General Motors (GM), however, that led to further active development in the mid 1990s.



Fig. 57.7 Fanuc's **IAD** for moving engine blocks [57.53]

Beginning in 1995, GM began to work with two groups: Kazerooni at University of California (UC) Berkeley and Colgate and Peshkin at Northwestern University. The Ford Motor Company also encouraged the participation of Fanuc Robotics, an established robotics supplier to the automotive industry. The automotive emphasis of the program helped steer the results toward applicability to general assembly (Sect. 57.1).

Fanuc developed a prototype gantry-type robot with force-input handles for human guidance, to be used at Ford to install truck instrument panels on a moving assembly line or to load cylinder heads on top of engine blocks. Fanuc's **IAD** possessed six powered admittance-controlled axes, like a conventional industrial robot. It was driven however by force inputs from a human user who could exert command forces on force-sensitive handles that moved with the end-effector. This is a key distinction from remote joystick or force-input devices that do not move with the payload. Operator motion together with the payload gives the operator immediacy to the task and to the motions of the robot, and even some haptic feedback from the task. In contrast, teleoperation produces distancing from the task that generally requires great cognitive effort to overcome, and leads to reduced speed. Additionally, a proximate operator can

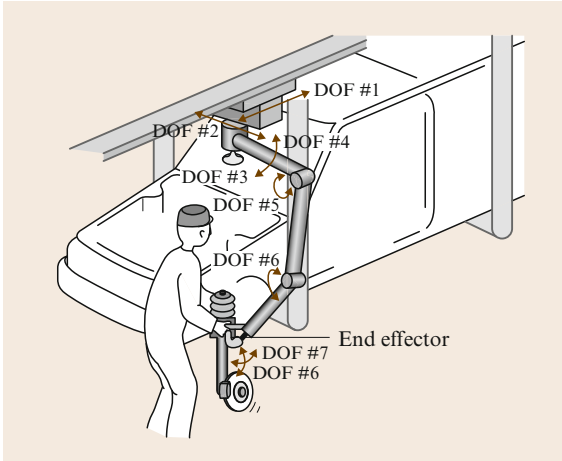


Fig. 57.8 Toyota's skill assist system [57.55]

detect and correct minor issues, such as a cable bundle that has draped the wrong way and needs to be moved. In addition the Fanuc IAD was able to implement virtual guides and walls, to be used as a *funnel* shape that assisted with moving a payload through an automobile door opening without collisions.

At the Toyota Motor Company, a similar concept, known as the skill assist, was developed at the end of the 1990s. A distinctive feature of this device was that the dynamic behavior of the assistive mechanical manipulator was varied according to the task phase. This approach was based on two observations. First, it was observed that the dynamic behavior preferred for moving a payload over long distances was primarily inertial, while the dynamic behavior preferred for precise positioning was primarily viscous. Second, it was observed that the phase of the task – start up, traversing, or fine positioning – could be automatically determined based on motion characteristics. Experiments performed with the skill assist showed significant improvements (reduction) in operator force (primarily for stopping), task completion time, and operator subjective impression.

Less structured types of IADs applications can be approached by the concept of human extenders [57.56], which emphasize amplification of human force and reach, such that for instance a human can grasp and lift a box that would otherwise be too wide and too heavy. Extenders may take the form of a robot arm or leg, or can even be worn by the user in the form of exoskeletons [57.9].

In industrial practice, it is much more common to support a payload from a cable or chain than from a rigid-link mechanism. Cables and chains have many



Fig. 57.9 pMRI in human power amplifiers [57.56]

advantages: they permit relatively free rotation about a vertical axis as well as small horizontal motions (or larger horizontal motions when supported by an articulated arm or crane), they have low inertia, low cost, and provide minimal obstruction. However, it is difficult to implement power assist with cable-based systems. Kazerooni built on his work with extenders to develop a servo-controlled lift assist that is marketed by Gobel, Inc. under the trade name G-Force. The key innovation of the G-Force was a slide handle inline with the cable that provided an intuitive input: sliding up to move up, and down to move down. Speed could be easily controlled.

Northwestern University's spin-off company Cobotics, Inc. (which was later acquired by the Stanley Assembly Technology division of StanleyWorks, Inc.) went a step further to develop the iLift and iTrolley modules that could be combined into a multiaxis human-assist crane. A key innovation was a cable angle sensor that could detect small departures of the cable from vertical, and use this to provide direction and speed commands to the iTrolleys, which provided motion in a horizontal plane in the manner of an overhead crane.

At this juncture it is worth noting that, while most of the IADs described so far have the structure illustrated

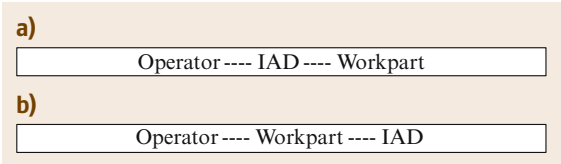


Fig. 57.10 (a) Structure in which the IAD is interposed between the operator and the workpart. (b) Structure allowing the operator to place hands directly on the workpart

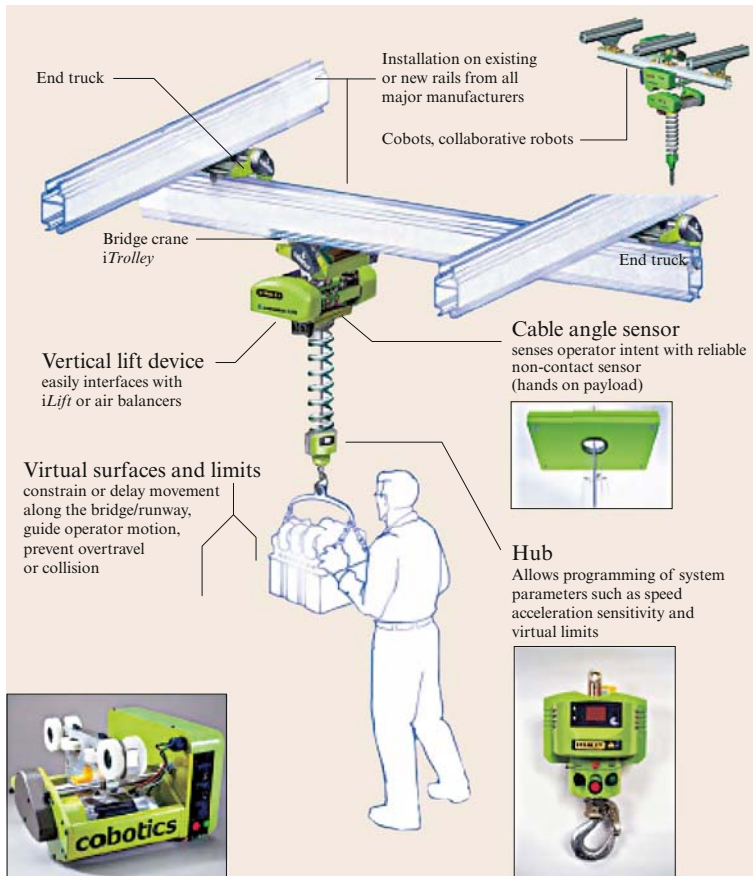


Fig. 57.11 The Stanley Cobotics modular system including lift assist, cable angle sensor, and powered trolleys

in Fig. 57.10a in which the IAD is interposed between the operator and workpart, cable-based systems are particularly amenable to the structure in Fig. 57.10b. Here, the operator manipulates the workpart directly while the IAD assists. The latter structure is often denoted *hands on payload*.

The iTrolley system illustrated in Fig. 57.11 is an example of *hands on payload*: the operator pushes on the workpart and the overhead crane follows. The strengths and weaknesses of this approach are evident. On the one hand, the operator is in intimate contact with the workpart, enabling highly intuitive movement. On the other hand, the cable can at best remain vertical at all times, contributing zero force to moving the workpart. If the cable were to *lead* the motion, the overall behavior would be unstable.

Nonetheless, *hands on payload* can provide a significant ergonomic advantage as the *pull test* data in Fig. 57.12 illustrate. Here, a subject pulled a variety of different payloads rapidly from a fixed starting

location to a fixed stopping location and his forces were recorded. Tests were run with traditional passive trolleys and with iTrolleys. The data show that the starting forces are modestly reduced with the iTrolley (because the iTrolley contributes to accelerating the overhead crane), and that the stopping forces are dramatically reduced (because the iTrolley eliminates overshoot of the overhead crane). This example illustrates an important point: IADs may be as valuable in moving the support structure as in moving the workpart itself.

Cobots – collaborative robots [57.13, 57, 58] – are a distinct approach to relieve humans from fatigue, stress, and injuries in manipulating heavy and/or awkward parts. Cobots presume a division of control between human and robot, with a robot perhaps supporting a payload and allowing a human to guide it, subject to constraint surfaces or virtual walls. Cobots are specifically designed to produce virtual guiding surfaces: software-defined paths or surfaces that constrain or direct the motion of the payload for the purpose of

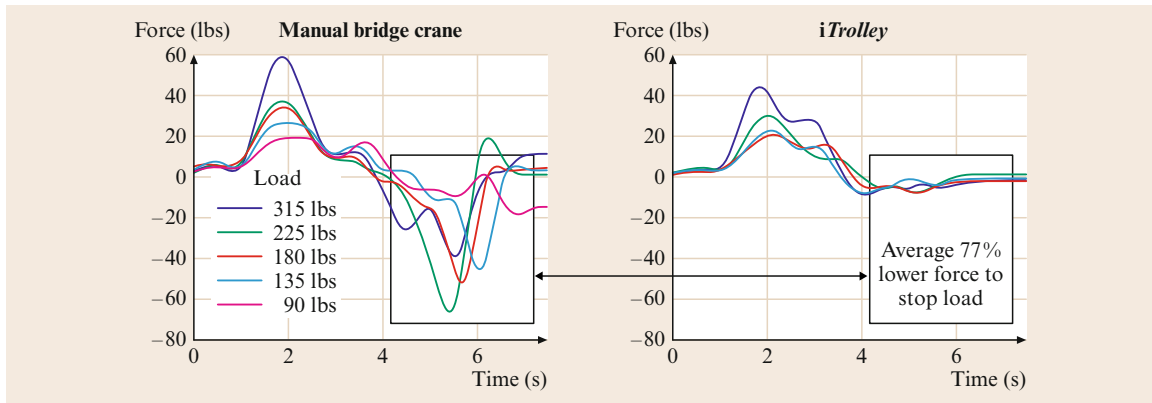


Fig. 57.12 Data from a set of pull tests showing the impact of the iTrolley and cable angle sensor on starting and stopping forces for various payloads

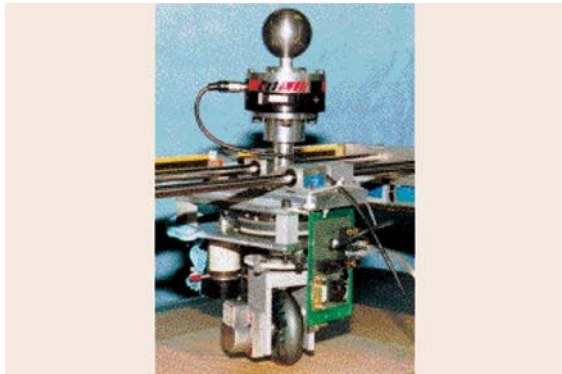


Fig. 57.13 The simplest cobot consists of a single rolling wheel on an xy frame [57.57]

guidance. Ergonomic as well as productivity benefits result from combining the strength and computer interface of the cobot with the sensing and dexterity of the human worker.

Consider the cobot shown in Fig. 57.13. A computer controls the steering of the wheel, thus confining the operator's handle to whatever virtual surfaces and free spaces are desired. In contrast to extenders and Fanuc's IADs, however, in the remaining direction – in this case the rolling path that the computer allows – the operator is in direct physical connection with payload; there is no sensor–actuator pair to interfere with the haptic sense between the payload and the operator. Additionally since computers only steer and do not drive the cobot forward, an element of inherent safety is introduced. The cobot concept has been extended to multiple degrees of



Fig. 57.14 A cobot assisting a human in assembling a car door

freedom: Fig. 57.14 shows a planar cobot, used for door handling at GM.

The cobot base permits planar motions on the floor (x , y , and rotation). The cobot indexes itself to the automobile's running board and guides the operator and payload (a door) along a path that brings the payload into proper alignment with the hinge axis of the automobile.

Cobots can also display the types of behavior mentioned above: free mode, in which the wheels are steered so as to comply with user forces; path mode, in which the cobot steers along a defined path through 3-space; and virtual surface mode in which the wheels are steered tangent to a software-defined constraint surface, resisting user forces that would violate the virtual surface.

57.5 Safety Standards for pHRI

In practice, standards are an important means of addressing and solving safety problems in the workplace, as well as encouraging the adoption of new and unfamiliar technology. Research work in the robotics industry and academia is both influenced by current work on standards, and has an influence on their evolution – which is particularly interesting at present.

The present landscape for robotics standards include well-established national standards (ANSI-RIA for the US, CSA in Canada, DIN in Germany, etc.), that are collected and harmonized by the International Organization for Standardization (ISO). Safety for industrial robots are addressed both by general standards on safety of machinery (e.g., ISO 12100:2003 *Basic concepts, general principles for design*; ISO 13849:1999 *Safety-related parts of control systems*; ISO 13855, *Positioning of protective equipment with respect to the approach speeds of parts of the human body*), and by robot-specific standards. The main ISO safety standard for robots is ISO 10218, which dates back to 1992 [57.6]. In 2002 a revision of the standard was undertaken. Work has been ongoing since, gradually turning what started as a simple harmonization effort into a genuine development effort introducing new and exciting concepts to the world of industrial robot safety [57.59]. The revised ISO 10218 (*Robots for industrial environment – Safety*) will be a two-part document. Part 1, entitled *Robot*, was published in 2006 [57.60]. Part 2, on which work has just begun, has a working title of *Robot System and Integration* and is not available at the time of writing. Part 2 is intended to address workplace safety requirements and is directed more to the end-user than the manufacturer.

The most salient changes of the 2006 standard revision are:

1. *new modes of operation*: the standard finally allows the introduction in the workplace of advanced robotics concepts, such as *simultaneous control* of multiple manipulators, as, e.g., in the master–slave configurations; *mobile* robots mounted on vehicles for industrial automation, and (most relevant to this chapter) *collaborative operation* in which purposely designed robots work in direct cooperation with a human within a defined workspace with the operator;
2. *control reliability*: while former standards placed reliance upon hardwired electromechanical components, the revised one allows *safety-rated soft-axis and space-limiting* control circuitry to use state-

of-the-art software, electronic, and network-based technology (including wireless);

3. *safeguarding and clearance*: instead of fixed safeguard distance, these can be evaluated based on the assessment of stopping time and distance to be provided by the robot manufacturer in different load conditions. In collaborative mode, hard limits on either the maximum dynamic power (80 W) or maximum static force (150 N) at the end-effector apply, as well as on its maximum velocity (250 mm/s).

Although the revision of ISO 10218 is already taking into consideration many more advanced features than in the past, evolution is still ongoing. In particular, in the domain of hands-on pHRI, the American National Standards Institute (ANSI) has established a committee, T-15, that has published a draft safety standard for intelligent assist devices (IADs) [57.61]. The committee defined IADs as *single- or multiple-axis devices that employ a hybrid, programmable, computer–human control system to provide human strength amplification, guiding surfaces or both*. Notable aspects of the standard include:

1. Risk assessments replace fixed rules: instead of declarations regarding how to accomplish safe operation, risk assessment procedures were advised for IAD and pHRI robotic technologies, to identify and mitigate risks in proportion to their seriousness and probability;
2. Safety-critical software: the greater complexity of human–robot interaction, and the observation that an abrupt power-down is not always a safe solution, necessitate a greater reliance on safety-critical software rather than safety-critical hardware. The T-15 draft standard requires that software- and firmware-based controllers shall, under any single component failure, lead to the shutdown of the system in a safe state, maintenance of a safe load position, and the prevention of subsequent automatic operation. For example, this degree of safety may be achieved by using microprocessor redundancy, microprocessor diversity, and self-checking;
3. Dynamic limits: physical limits of people are taken into account by requiring that operators be able to *outrun, overpower, or turn off IADs*. Speed must not exceed 2.0 m/s, a fast walk; overforce or overload devices or techniques must be used that can reliably detect an impulse force of 267 N (60 lbf) (the committee dictated that a human operator should win

a fight). These and similar limits could be further reduced to the extent practical as determined by a risk assessment;

4. Emergency stops: sole reliance on the traditional red mushroom e-stop button was felt to create a hazard. For instance, in some unexpected situations one might want an IAD to continue to actively *track* a moving vehicle, rather than come immediately to a halt and possibly drag a part and a person engaged in a moving line. The T-15 draft standard demands a traditional e-stop, but also permits *an IAD may have one or more context-based safety stop*

circuits. When used, inputs should be provided to allow application-specific external devices to initiate context-based safety stops;

5. Man-machine interface: IADs may operate in different modes (free mode, hands-on-controls, hands-on-payload, line tracking, etc.). The T-15 committee found that *mode misunderstanding* was a likely cause of safety problems. IADs should have few modes, well understood by their operators, well communicated to the operator (*what mode is this IAD in?*) and well commanded by the operator to the IAD (*go into free mode!*).

57.6 Conclusions

The next generation of robots will coexist with humans and will interact with us physically. In this chapter we have illustrated some of the motivations and economic factors that are pushing this revolution forward. A robot arm that is to interact with humans must place safety at a premium as a design consideration. As Asimov's laws of robotics have it 'A robot

may not injure a human being...', or, to rephrase the science fiction author, under no circumstances should a robot cause harm to people, directly or indirectly, in regular operation or in failures. Another requirement on robot manipulators remains their performance: their accuracy and speed in performing tasks as required.

References

- 57.1 OSHA: *Industrial Robots and Robot System Safety*, Section IV, Chapter 4, OSHA Technical Manual (Occupational Safety and Health Administration, Washington 2006) http://www.osha.gov/dts/osta/otm/otm_iv/otm_iv_4.html
- 57.2 K. Bixby: Proactive safety design and implementation criteria, Proc. National Robot Safety Conf. (1991)
- 57.3 B.S. Dhillon: *Robot Reliability and Safety* (Springer, New York 1993)
- 57.4 J.H. Graham (Ed.): *Safety, Reliability, and Human Factors in Robotic Systems* (Van Nostrand Reinhold, New York 1991)
- 57.5 ANSI: *American National Standard for Industrial Robots and Robot Systems – Safety Requirements*, ANSI/RIA R15.06-1986 (American National Standards Institute, New York 1986)
- 57.6 ISO 10218: *Manipulating Industrial Robots – Safety* (International Organization for Standardization, Geneva 1992)
- 57.7 OSHA: *Guidelines for Robotic Safety*, OSHA Instruction PUB 8-1.3 (Occupational Safety and Health Administration, Washington 1987)
- 57.8 USDL: *Lost-worktime Injuries and Illnesses: Characteristics and Resulting Time Away From Work*, USDL 05-2312 (US Department of Labor, Washington 2005) <http://www.bls.gov/iif/home.htm>
- 57.9 A. Chu, H. Kazerooni, A. Zoss: On the biomimetic design of the Berkeley lower extremity exoskeleton (BLEEX), Proc. IEEE Int. Conf. Robot. Autom. (2005) pp. 4345–4352
- 57.10 M. Bergamasco, A. Frisoli: A new force-feedback arm exoskeleton for haptic interaction in virtual environments, World Haptics Conf. WHC (Pisa 2005) pp. 195–201
- 57.11 J. Guiochet, A. Vilchis: Safety analysis of a medical robot for tele-echography, 2nd IARP IEEE/RAS Joint Workshop Tech. Challenge Dependable Robots Human Env. (Toulouse 2002)
- 57.12 T. Noritsugu, T. Tanaka, T. Yamanaka: Application of rubber artificial muscle manipulator as a rehabilitation robot, Proc. IEEE Int. Workshop Robot Human Commun. (1996) pp. 112–117
- 57.13 M. Peshkin, J.E. Colgate, W. Wannasupphorasit, C. Moore, B. Gillespie: Cobot architecture, IEEE Trans. Robot. Autom. **17**(4), 377–390 (2000)
- 57.14 R. Horowitz, P. Li, J. Shields: Control of self-optimizing exercise machines, IFAC Ann. Rev. Contr. **24**(1), 201–213 (2000)
- 57.15 K. T. Ulrich, T. T. Tuttle, J. P. Donoghue, W.T. Townsend: *Intrinsically Safer Robots*, Final Report for NASA Contract #NAS10-12178 (Barrett Technology Inc., Cambridge 1995)

- 57.16 G. Giralto, F. Ingrand: Safely among us – dependability in human-friendly robots, IEEE-RAS Mag. **11**(2), 1–15 (2004), Special Issue
- 57.17 K. Ikuta, H. Ishii, M. Nokata: Safety evaluation method of design and control for human-care robots, Int. J. Robot. Res. **22**(5), 281–297 (2003)
- 57.18 A. Albu-Schäffer, A. Bicchi, G. Boccadamo, R. Chatila, A. De Luca, A. De Santis, G. Giralto, G. Hirzinger, V. Lippiello, R. Mattone, R. Schiavi, B. Siciliano, G. Tonietti, L. Villani: Physical human-robot interaction in anthropic domains: Safety and dependability, 4th IARP/IEEE-EURON Work. Tech. Challenges Dependable Robots Human Env. (Nagoya 2005)
- 57.19 R. Piggin: A new approach to robotic safety: SafetyBUS p at BMW, Ind. Robot Int. J. **29**(6), 524–529 (2002), MCB UP Ltd.
- 57.20 C.W. Gadd: Use of weighted impulse criterion for estimating injury hazard, 10th Stapp Car Crash Conf. (New York 1966)
- 57.21 J. Newman, N. Shewchenko, E. Welbourne: A proposed new biomechanical head injury assessment function – the maximum power index, Stapp Car Crash J. **44**, 215–247 (2000)
- 57.22 J. Versace: A review of the severity index, Proc. 15th Stapp Car Crash Conf. (New York 1971)
- 57.23 M. Zinn, O. Khatib, B. Roth, J.K. Salisbury: A new actuation approach for human-friendly robot design, Int. Symp. Exp. Robot. (S. Angelo d’Ischia, 2002)
- 57.24 A. Bicchi, G. Tonietti: Fast and soft arm tactics, IEEE Robot. Autom. Magazine **11**(2), 22–33 (2004), on Dependability of Human-Friendly Robots
- 57.25 S. Haddadin, A. Albu-Schäffer, G. Hirzinger: Safety evaluation of physical human-robot interaction via crash-testing, Robot. Sci. Syst. Conf. (RSS2007) (Atlanta 2007)
- 57.26 USD National Highway Traffic Safety Administration: *Actions to Reduce the Adverse Effect of Airbags*, FMVSS No. 208 (US Dept. of Transportation, Washington 1997)
- 57.27 E. Cheung, V. Lumelsky: Proximity sensing in robot manipulator motion planning: system and implementation issues, IEEE Trans. Robot. Autom. **5**(6), 740–751 (1989)
- 57.28 H. Iwata, H. Hoshino, T. Morita, S. Sugano: Force detectable surface covers for humanoid robots, Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatron. (2001) pp. 1205–1210
- 57.29 W.T. Townsend: The Effect of Transmission Design on Force-Controlled Manipulator Performance. Ph.D. Thesis (Massachusetts Institute of Technology, Cambridge 1988) Also published as an MIT Artificial Intelligence Laboratory Technical Report, AI-TR #1054 (Cambridge 1988)
- 57.30 K. Salisbury, W. Townsend, B. Eberman, D. DiPietro: Preliminary design of a whole-arm manipulation system (wams), Proc. IEEE Int. Conf. Robot. Autom. (Philadelphia 1988) pp. 254–260
- 57.31 G. Hirzinger, A. Albu-Schäffer, M. Hahnle, I. Schäffer, N. Sporer: On a new generation of torque controlled light-weight robots, IEEE Int. Conf. Robot. Autom. (2001)
- 57.32 P. Kazanzides, J. Zuhars, B. Mittelstadt, R.H. Taylor: Force sensing and control for surgical robot, Proc. Int. Conf. Robot. Autom. (1992)
- 57.33 J. Heinzmann, A. Zelinsky: Quantitative safety guarantees for physical human-robot interaction, Int. J. Robot. Res. **22**(7/8), 479–504 (2003)
- 57.34 A. Albu-Schäffer, G. Hirzinger: Cartesian compliant control strategies for light-weight, flexible joint robots. In: *Control Problems in Robotics*, Springer Tracts in Advanced Robotics – STAR, Vol. 4, ed. by A. Bicchi, H.I. Christensen, D. Prattichizzo (Springer, Berlin, Heidelberg 2003)
- 57.35 A. De Luca, A. Albu-Schäffer, S. Haddadin, G. Hirzinger: Collision detection and safe reaction with the DLR-III lightweight manipulator arm, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (Beijing 2006)
- 57.36 R. Kelly, R. Ortega, A. Ailon, A. Loria: Global regulation of flexible joints robots using approximate differentiation, IEEE Trans. Autom. Contr. **39**(6), 1222–1224 (1994)
- 57.37 B. Siciliano, L. Villani: An inverse kinematics algorithm for interaction control of a flexible arm with a compliant surface, Contr. Eng. Prac. **9**, 191–198 (2001)
- 57.38 A. De Luca: Feedforward/feedback laws for the control of flexible robots, Proc. IEEE Int. Conf. Robot. Autom. (2000) pp. 233–240
- 57.39 G.A. Pratt, M. Williamson: Series elastics actuators, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (1995) pp. 399–406
- 57.40 M. Okada, Y. Nakamura: Development of the cybernetic shoulder – a three-DOF mechanism that imitates biological shoulder motion, Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst. (1999) pp. 453–548
- 57.41 B. Brogliato, R. Lozano: Correction to adaptive control of robot manipulators with flexible joints, IEEE Trans. Autom. Contr. **AC-41**(6), 920–922 (1996)
- 57.42 M. Zinn, O. Khatib, B. Roth, J.K. Salisbury: A new actuation approach for human-friendly robot design, Int. J. Robot. Res. **23**(4/5), 379–398 (2005)
- 57.43 A. Sharon, N. Hogan, D.E. Hardt: High bandwidth force regulation and inertia reduction using a macro/micro manipulator system, Proc. IEEE Int. Conf. Robot. Autom. (1988) pp. 126–132
- 57.44 N. Hogan: Adaptive control of mechanical impedance by coactivation of antagonist muscles, IEEE Trans. Autom. Contr. **29**(8), 681–690 (1984)
- 57.45 J.L. Nevins, D.E. Whitney: The Force Vector Assembler Concept, Int. Conf. Robot. Manip. (Udine 1973)
- 57.46 T. Morita, S. Sugano: Design and development of a new robot joint using a mechanical impedance

- adjuster, Proc. IEEE Int. Conf. Robot. Autom. (1995) pp. 2469–2475
- 57.47 A. Bicchi, S.L. Rizzini, G. Tonietti: Compliant design for intrinsic safety: General issues and preliminary design, IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2001)
- 57.48 K. Suita, Y. Yamada, N. Tsuchida, K. Imai, H. Ikeda, N. Sugimoto: A failure-to-safety *kyozon* system with simple contact detection and stop capabilities for safe human-autonomous robot coexistence, IEEE Int. Conf. Robot. Autom. (1995) pp. 3089–3096
- 57.49 Y. Yamada, Y. Hirasawa, S. Huang, Y. Uematsu, K. Suita: Human-robot contact in the safeguarding space, IEEE/ASME Trans. Mechatron. **2**(4), 230–236 (1997)
- 57.50 F. Caccavale, L. Villani (Eds.): *Fault Diagnosis and Fault Tolerance for Mechatronic Systems: Recent Advances*, Springer Tracts in Advanced Robotics, Vol. 1 (Springer, Heidelberg 2003)
- 57.51 S. Morinaga, K. Kosuge: Collision detection system for manipulator based on adaptive impedance control law, IEEE Int. Conf. Robot. Autom. (2003) pp. 1080–1085
- 57.52 A. De Luca, R. Mattone: Sensorless robot collision detection and hybrid force/motion control, IEEE Int. Conf. Robot. Autom. (2005) pp. 1011–1016
- 57.53 H. Forcinio: *Intelligent Assist Devices Coming of Age* (RIA, Ann Arbor 2006), Robotics Online <http://www.roboticsonline.com/public/articles/archivedetails.cfm?id=212>
- 57.54 R.S. Mosher: From handiman to hardiman, Trans. Soc. Autom. Eng. **16**, 588–597 (1967)
- 57.55 Y. Yamada, H. Konosu, T. Morizono, Y. Umetani: Proposal of Skill-Assist: a system of assisting human workers by reflecting their skills in positioning tasks, IEEE Int. Conf. Syst. Man Cybern. (1999), Toyota Technol. Inst., Nagoya
- 57.56 H. Kazerooni: The human power amplifier technology at the University of California, Berkeley, J. Robot. Autom. Syst. **19**, 179–187 (1996)
- 57.57 J.E. Colgate, W. Wannasuphprasit, M.A. Peshkin: Cobots: Robots for collaboration with human operators, Proc. ASME Dyn. Syst. Cont. Div., Vol. 58 (1996) pp. 433–440
- 57.58 K.M. Lynch, C. Liu, A. Sorensen, S. Kim, M. Peshkin, J.E. Colgate, T. Tickel, D. Hannon, K. Shiels: Motion guides for assisted manipulation, Int. J. Robot. Res. **21**(1), 27–43 (2002)
- 57.59 J. Fryman: Director, Standards Development, RIA. Recent Developments in Robot Safety Standards, Robotics Online (2003)
- 57.60 ISO2018-1: *Robots for industrial environments – Safety requirements. Part 1: Robot* (International Organization for Standardization, Geneva 1992)
- 57.61 Robotic Industries Association: T15.1 Draft standard for trial use for intelligent assist devices – personnel safety requirements (2002)

58. Social Robots that Interact with People

Cynthia Breazeal, Atsuo Takanishi, Tetsunori Kobayashi

This chapter surveys some of the principal research trends in social robotics and its application to human–robot interaction (HRI). Social (or sociable) robots are designed to interact with people in a natural, interpersonal manner – often to achieve social–emotional goals in diverse applications such as education, health, quality of life, entertainment, communication, and collaboration. The long-term goal of creating social robots that are competent and capable partners for people is quite a challenging task. They will need to be able to communicate naturally with people using both verbal and nonverbal signals. They will need to engage us not only on a cognitive level, but on an emotional level as well. They will need a wide range of social–cognitive skills and a theory of other minds to understand human behavior, and to be intuitively understood by people. A deep understanding of human intelligence and behavior across multiple dimensions (i. e., cognitive, affective, physical, social, etc.) is necessary in order to design robots that can successfully play a beneficial role in the daily lives of people. This requires a multidisciplinary approach where the design of social robot technologies and methodologies are informed by robotics, artificial intelligence,

58.1 Social Robot Embodiment.....	1350
58.2 Multimodal Communication	1352
58.2.1 Robots that Express Paralinguistic Information.....	1352
58.2.2 Robots that Understand Paralinguistic Information.....	1353
58.2.3 Group Conversation	1355
58.2.4 Communication in Collaboration ...	1355
58.3 Expressive Emotion–Based Interaction....	1356
58.3.1 Kismet: Inspiration from Developmental Psychology ...	1356
58.3.2 WE-4RII: A Model of Emotion Inspired by Motion	1357
58.4 Socio–cognitive Skills	1360
58.4.1 Shared Attention.....	1360
58.4.2 Emotional Empathy	1362
58.4.3 Mental Perspective Taking	1362
58.4.4 Perspective Taking in Collaboration	1363
58.5 Conclusion and Further Reading	1365
References	1366

chology, neuroscience, human factors, design, anthropology, and more.

The way a person interacts with a social robot (or sociable robot) is quite different from interacting with the majority of autonomous mobile robots today. Modern autonomous robots are generally viewed as tools that human specialists use to perform hazardous tasks in remote environments (i. e., sweeping minefields, inspecting oil wells, mapping mines, etc.). In dramatic contrast, social (or sociable) robots are designed to engage people in an interpersonal manner, often as partners, in order to achieve social or emotional goals.

The development of socially intelligent and socially skillful robots drives research to develop autonomous or semiautonomous robots that are natural and intuitive for the general public to interact with, communicate with, work with as partners, and teach new capabilities. *Dautenhahn's* work is among the earliest in thinking about robots with interpersonal social intelligence where relationships between specific individuals are important [58.1,2]. These early works pose the question *What are the common social mechanisms of communication and understanding that can produce efficient, enjoyable,*

natural, and meaningful interactions between humans and robots?

Promisingly, there have been initial and ongoing strides in all of these areas ([58.3–11], etc.). In addition, this domain motivates new questions for robotics researchers, such as how to design for a successful long-term relationship where the robot remains appealing and provides consistent benefit to people over weeks, months, and even years. The benefit that social robots provide people extends far beyond strict task performing utility to include educational (Chap. 55), health and therapeutic (Chap. 53), domestic (Chap. 54), social and emotional goals (e.g., entertainment, companionship, communication, etc.), and more.

58.1 Social Robot Embodiment

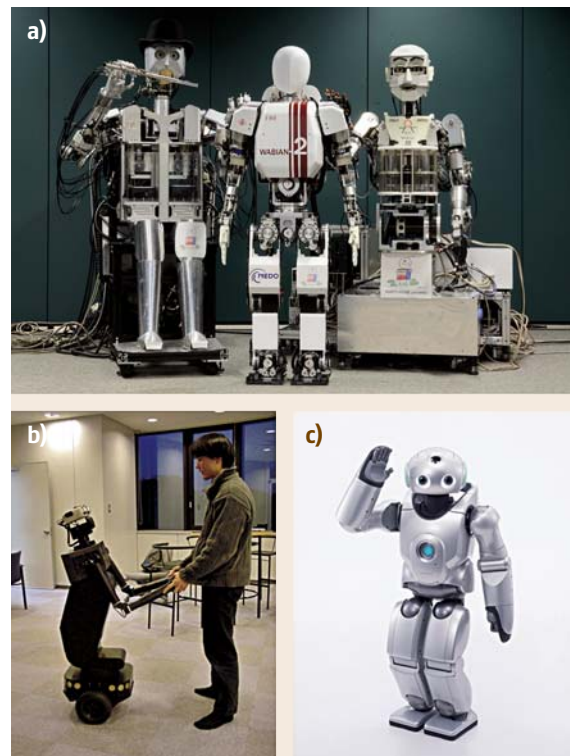
Social robots are designed to interact with people in human-centric terms and to operate in human environments alongside people. Many social robots are humanoid or animal-like in form, although this does not have to be the case. A unifying characteristic is that social robots engage people in an interpersonal manner, communicating and coordinating their behavior with humans through verbal, nonverbal, or affective modalities. As can be seen in the following examples, social robots exploit many different modalities to communicate and express social-emotional behavior. These include whole-body motion, proxemics (i.e., interpersonal distance), gestures, facial expressions, gaze behavior, head orientation, linguistic or emotive vocalization, touch-based communication, and an assortment of display technologies.

For social robots to close the communication loop and coordinate their behavior with people, they must also be able to perceive, interpret, and respond appropriately to verbal and nonverbal cues from humans.

Fig. 58.1a–c Examples of socially interactive humanoid robots: **(a)** humanoid robots developed at Waseda University from *left to right*: a flautist robot WF-4RII [58.12], **WABIAN-2** [58.13], and WE-4RII [58.14] **(b)** Robovie III developed by the Advanced Telecommunications Research Institute International (ATR) Intelligent Robotics and Communication Laboratories, which is able to gesture with its arms and give a hug **(c)** quest for curiosity (QRIO), a small biped entertainment robot previously developed by Sony, is very well known for its impressive dancing ability

We begin this chapter with a brief overview of a wide assortment of socially interactive robots that have been developed around the world (Sect. 58.1). We follow with selected topics that highlight some of the representative research themes: multimodal communication (Sect. 58.2), expressive emotion-based interaction (Sect. 58.3), and social-cognitive skills (Sect. 58.4). We rely on examples from our own research programs to illustrate these trends, while making reference to other excellent work performed in other research labs. The following robotic platforms are used as case studies through the sections: Kismet and Leonardo developed at MIT, and Waseda Eye No. 4 Refined II (WE-4RII), ROBISUKE, and ROBITA developed also at Waseda University.

Given the richness of human behavior and the complexity of human environments, many social robots are among the most sophisticated, articulate, behaviorally rich, and intelligent robots today.



As shown in Fig. 58.1, a number of socially interactive humanoid robots have been developed (see Chap. 56) that can participate in whole body social interaction with people such as dancing [58.15], walking hand-in-hand [58.16,17], playing a musical duet [58.12], or transferring skills to unskilled persons [58.18]. Their arms and hands are designed to exhibit human-like gestures such as pointing, shrugging shoulders, shaking hands, or giving a hug [58.19–21]. Some of them are designed with mechanical faces to communicate with humans via facial expressions.

Whereas many of these humanoids have a mechanical appearance, android robots are designed to have a very human-like appearance with skin, teeth, hair, and clothes (Fig. 58.2). A design challenge of android robots is to avoid the *uncanny valley* where the appearance and movement of the robot resemble more of an animate corpse than a living human. Designs that fall within the *uncanny valley* elicit a strong negative reaction from people [58.30].

There are a number of more creature-like social robots that take their aesthetic and behavioral inspiration from animals (Fig. 58.3). Given that people pet and stroke companion animals, touch-based communication has been explored in several of these more animal-inspired robots. Sony's entertainment robot dog, AIBO [58.26], is a well-known commercial example. Other robots in this category have a more organic appearance, such as the therapeutic companion robot seal, Paro [58.27]. Researchers have also chosen to design robots with a more fanciful appearance, melding anthropomorphic with animal-like qualities such as Leonardo ([58.28, 29, 31], etc.).

Many social robots are not overtly humanoid or zoomorphic, but still capture key social attributes (Fig. 58.4), for instance, one of the best-known and pioneering social robots Kismet [58.3] developed at the MIT Artificial Intelligence Lab. Kismet had a very expressive mechanical face with anthropomorphic features like large blue eyes. Another example is the dancing robot Keepon developed by National Institute of Information and Communications Technology (NiCT) (Japan). This small yellow robot has a simplistic face and uses a classic animation technique called *squash and stretch* for expression of the body [58.32].

Many mobile social robots have been fitted with faces to enhance social interaction (Fig. 58.4). Some examples are the elder-care robot, Pearl, and the robotic receptionist Valerie with a graphical face on a liquid-crystal display (LCD) screen [58.33], both developed at Carnegie Mellon University [58.34]. Other examples are

robots like PaPeRo developed by NEC, aiming at a commercial product [58.35]. Still, some social robots have

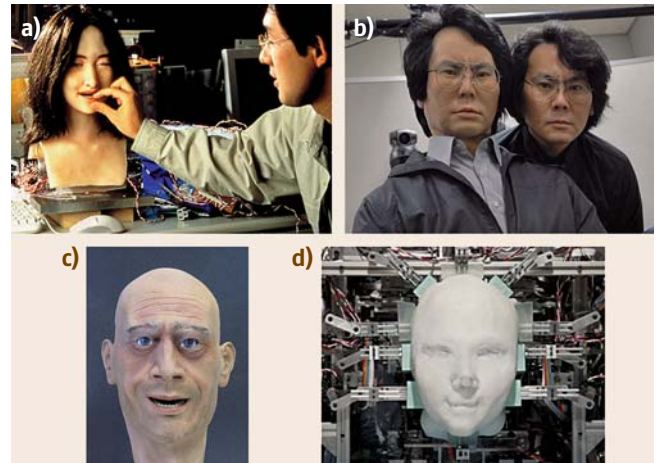


Fig. 58.2a–d Some examples of androids: (a) one of the earliest face robots developed at the Science University of Tokyo [58.22] (b) Geminoid HI-1 developed by the ATR Intelligent Robotics and Communication Laboratories [58.23] (c) ROMAN, developed at the University of Kaiserslautern [58.24] (d) the Waseda DOCOMO FACE Robot No. 2 developed by Waseda University [58.25]

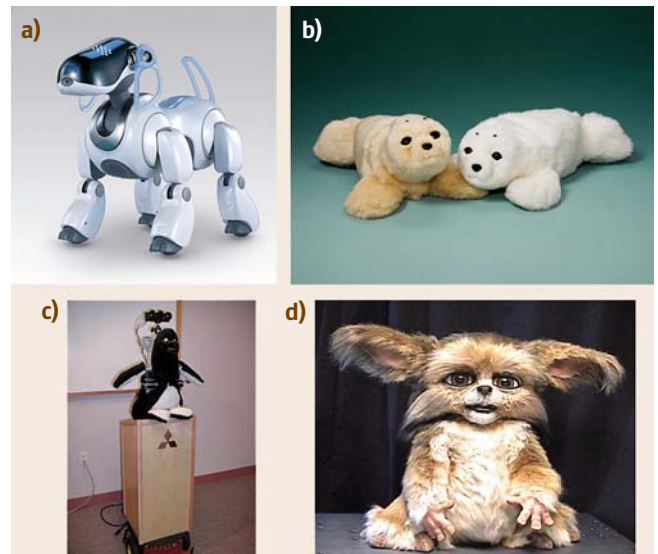


Fig. 58.3a–d Examples of social robots inspired by animals with anthropomorphic qualities (from left to right): (a) AIBO, the robotic dog previously developed by Sony [58.26], (b) Paro the therapeutic seal robot developed at AIST [58.27], (c) Mel the conversational robotic penguin developed at MERL [58.28], and (d) Leonardo developed at the MIT Media Lab [58.29]

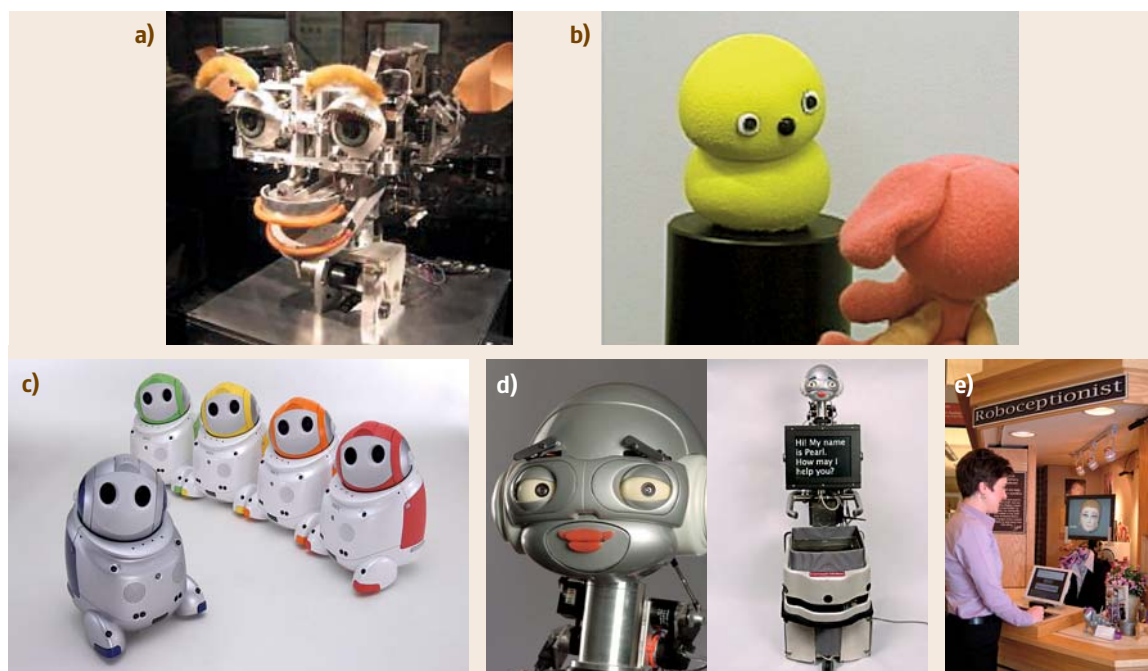


Fig. 58.4a–e Examples of social robots that are neither humanoid nor zoomorphic but capture key social attributes: (a) Kismet [58.3], (b) Keepon [58.32], (c) PaPeRo [58.35], (d) Pearl [58.34], (e) Valerie [58.33]

no overt social features like faces or eyes, but rely purely on language-based communication. Issues of proxemics on mobile social robots have also been explored such as

how a robot should approach a person [58.36], follow a person [58.37], or maintain appropriate interpersonal distance [58.38].

58.2 Multimodal Communication

Natural conversational ability is an important skill for social robots. Historically, even first-generation humanoid robots developed in the 1970s and 1980s (i.e., WABOT and WABOT-2) had primitive conversational skills modeled as simple combinations of speech input/output mappings [58.39, 40]. More recent examples are PaPeRo [58.35] and receptionist robot (ASKA) [58.41] that have conversational functions to work as information terminals.

In natural human conversation, however, people send and receive nonverbal information that supplements linguistic information. These paralinguistic cues help smooth and regulate communication between individuals.

The representative roles of paralinguistic information are as follows:

1. *Regulators*: expressions such as gestures, poses, and vocalizations that are used to regulate/control conversational turn-taking.
2. *State displays*: signs of internal state including affect, cognitive, or conversational states that improve interface transparency.
3. *Illustrators*: gestures that supplement information for the utterance. These include pointing gestures, iconic gestures,

This section focuses on conversational robots with paralinguistic communication abilities. Modern social robots employ both linguistic and paralinguistic information to perform different kinds of tasks [58.28, 29, 33].

58.2.1 Robots that Express Paralinguistic Information

In many cases, the same paralinguistic information can be conveyed through auditory or visual channels. However the characteristics of these channels have different properties. Sound has a strong property of attracting attention instantaneously. It can be used effectively for interruption, but it can also be disruptive and annoying. In addition, the timing of auditory paralinguistic cues is very strict. Overall, auditory paralinguistic signals are not suitable for continuous display. Visual paralinguistic cues are silent, the timing is a slightly less critical, and they can be used continuously. Hence, auditory and visual cues can be effectively used together to cooperatively convey the same information and emphasize it, or they can be used to convey different information simultaneously and contribute to efficiency. Thus, it is very important to choose the proper combination of modality and paralinguistic cues according to the situation. We provide examples below.

Regulatory Cues

Some of the earliest social robots displayed paralinguistic information to regulate interaction with people. Hadaly2 was the first robot to use mutual gaze as a paralinguistic cue to regulate conversation [58.42, 43]. When the robot and human achieved mutual gaze, approximated using face recognition to determine when the human's face was facing the robot, Hadaly2 expressed readiness to commence conversation by blinking its eyes. Other examples are *Kismet* [58.44] and *Leonardo* [58.29], which have implemented paralinguistic cues called *envelope displays* to regulate the exchange of speaking turns. Humans tend to make eye contact and raise their eyebrows when ready to relinquish their speaking turn, and tend to break gaze and blink when starting their speaking turn. These cues were shown to be effective in smoothing and synchronizing the exchange of speaking turns with human subjects, resulting in fewer interruptions and awkward long pauses between turns [58.3, 45].

State-Display Cues

Other robots use state-display cues whereby the face or gaze of the robot is used to indicate its conversational or cognitive state. In general, this makes the robot's internal state more transparent to a person, so they can better predict and interpret the robot's conversational state and level of understanding. ROBITA used the tightness of its facial expression to indicate readiness to engage in

conversation; a tight face was used to express conversational readiness, while a loose face communicated a lack of readiness to engage [58.5, 46].

Other state-display cues are back-channel responses. Human listeners use back-channel feedback (such as small head nods) to convey when speakers are successfully following the conversation. Response time for back-channel cues is very important because that cue is associated with the corresponding content. ROBISUKE [58.47] employed finite-state transducer (FST) technology to achieve rapid understanding of the speech signal. This allowed the robot to resolve ambiguities in meaning and prepare its own responses even when the speaker was in mid-sentence. ROBITA [58.5] provided humans with back-channel information via head nods and a tight face expression while listening.

Another back-channel signal is an expression of confusion by the listener (verbal or nonverbal). This flags the speaker to stop and try to repair the broken communication. Robots such as Leonardo and ROBITA use facial displays of confusion when speech recognition fails in order to intuitively communicate to the human that he or she should repeat their last utterance.

Similarly, gaze direction is a highly salient cue to convey the attentional focus of a robot. This is very useful if a human is trying to point out a particular object as a shared referent – such as pointing to an object before labeling it for the robot. These paralinguistic cues are useful for facilitating efficient conversational progress whereby errors and misunderstandings are identified and repaired immediately [58.45]; for instance, Leonardo's gaze behavior is used for performing envelope displays as well as for establishing shared attention with the human partner. Human subject studies have verified that Leonardo's paralinguistic cues contribute positively to the transparency of the robot's behavior and make the overall interaction more robust and efficient [58.45].

Illustrator Cues

A number of robots have implemented *illustrator* cues to direct the attention of a human [58.28, 48]. Often these robots use a variety of cues and the timing between them (such as gaze, head pose, pointing gestures, and conversational speech) to perform joint attention. In cases where a robot may be interacting with more than one person, the robot must properly take into account the location and orientation of the object, itself, and the individuals. ROBITA used such information to choose the proper gesture while considering the other people's point of view.

58.2.2 Robots that Understand Paralinguistic Information

Humans readily express paralinguistic cues when interacting with robots just as they do with people. Consequently, conversational robots must be able to recognize and properly respond to these cues as well. This is a very difficult research challenge given the wide variety, subtlety, and timing of these human cues.

Regulatory Cues

A few robots can track a human's paralinguistic cues to help regulate the conversation. The most common cues used for recognizing the end of the human's speaking turn are mutual gaze (estimated using head pose to determine when human looks back to the robot) and paused speech. As a more sophisticated example, humans frequently provide short acknowledgement utterances (e.g., 'uh-huh', 'um-hmm', 'huh', etc.) as the robot explains something. These responses are either acknowledgments or acknowledgment-like repetitions, or

ask-backs or ask-back-like repetitions. It is very difficult to distinguish these two kinds of utterances from the linguistic information as represented by the transcription of the utterance. The only way to distinguish them is by their prosody (not what, but how something is said). ROBISUKE could distinguish the utterance as either an acknowledgment or an ask-back from the prosody of utterance [58.49].

State-Display Cues

A number of robots are able to recognize and respond to state-display cues such as back-channel feedback nods, acknowledgement of an utterance, and attentional focus. One of the most robust systems for handling back channel feedback nods is Mel [58.28]. A sophisticated head nod recognition system was developed whereby the robot could successfully distinguish small feedback nods from other kinds of head nods such as those that communicate agreement. Mel used this information to determine its own nodding behavior in order to be an appropriate response for the human. In a series of human subject studies, *Sidner* et al. found these paralinguistic cues to enhance the social engagement of the robot to people [58.28]. With respect to recognizing successful or unsuccessful acknowledgement of an utterance, ROBISUKE used facial expression and prosody of the person's utterance [58.50] to make this determination. In a collaborative assembly task scenario, *Sakita* et al. [58.51] presented a robotic system that used human gaze information to deduce the human's intention of which object to operate upon next. The robot used this information to choose an appropriate cooperative action such as either taking over for a human, settling a human's hesitation, or executing simultaneously with a human.

Illustrator Cues

A number of robots are able to recognize deictic gestures of a person conveyed either through pointing gestures or head pose. For example, Leonardo is able to infer the object referent in an interaction by considering a number of factors including pointing gesture, head pose, and speech. *Brooks* and *Breazeal* [58.52] developed a deictic recognition system that enabled a robot to infer the correct object referent from correlated speech and deictic gesture. Interestingly, it was found that the accuracy of the human's pointing gesture is surprisingly poor. As a result, the deictic recognition system relies on coordinated speech and gesture information, with spatial knowledge provided by a three-dimensional (3-D) spatial database constructed by the robot using real-time vision, and a deictic spatial reasoning system. This sys-

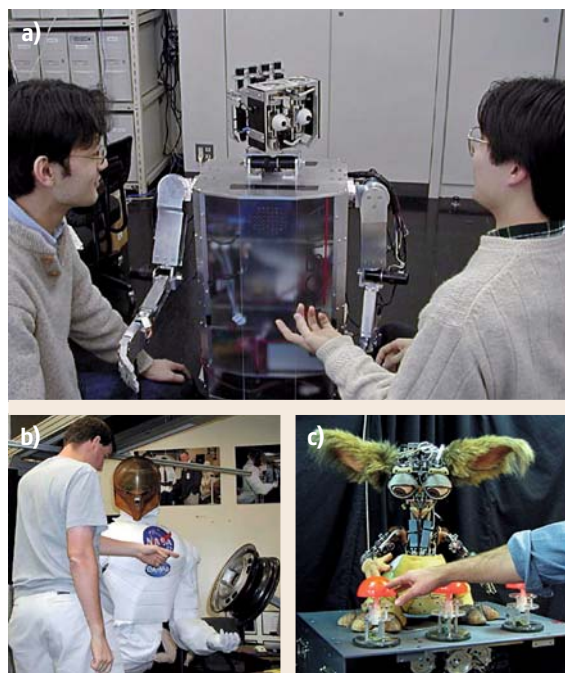


Fig. 58.5a–c Examples of conversational robots: (a) ROBITA performing group conversation; (b) Robonaut interpreting the pointing gestures of a human to determine which nut to fasten on the wheel; (c) Leonardo uses gaze and joint attention to ground the human's pointing gesture for the desired referent

tem was successfully demonstrated on the dexterous humanoid Robonaut developed at National Aeronautics and Space Administration (NASA) Johnson Space Center (JSC) (Fig. 58.5b) where the human points to and labels a set of four bolts on a wheel to be fastened in order by the robot.

58.2.3 Group Conversation

The ability to express and understand paralinguistic cues plays an important role in face-to-face conversation. The same is true for group conversation where a robot converses with two or more people. We use ROBITA, shown in Fig. 58.5a, as the illustrative example of this ability [58.46].

To continue a conversation, it is important for all conversational participants to understand who plays which role. ROBITA frames this problem with respect to information flow – to understand who is speaking to whom and when, and to determine each person’s presence as a conversational partner. During group conversation, ROBITA tries to classify the participants as the speaker or as listeners. There is only one speaker at a time and the rest are listeners. The listeners are classified into a primal listener, to whom the utterance is directed, and secondly listeners, who observe the message exchange between the speaker and the primal listener. ROBITA discerns these roles by recognizing the face direction of the speaker. The person to whom the speaker is looking is recognized as the primal listener.

To convey participation and improve its social presence in conversation, ROBITA tries to understand the conversational roles to look at the appropriate person. If the speaker faces ROBITA, then the robot recognizes itself as the primal listener and that the message is intended for itself. ROBITA faces the speaker when it is the primal listener. When ROBITA is a secondary listener, it looks at either the speaker or the primal listener.

58.2.4 Communication in Collaboration

Verbal and paralinguistic communication plays a very important role in coordinating joint action during collaborative tasks. Sharing information through communication acts is critical given that each teammate often has only partial knowledge relevant to solving the problem, each has different capabilities, and possibly diverging beliefs about the state of the task. For instance, all teammates need to establish and maintain a set of mutual beliefs regarding the current state of the task, the

respective roles and capabilities of each member, and the responsibilities of each teammate [58.53]. This is called *common ground* [58.54].

Dialog certainly plays an important role in establishing common ground. Each conversant is committed to the shared goal of establishing and maintaining a state of mutual belief with the other. To succeed, the speaker composes a description that is adequate for the purpose of being understood by the listener, and the listener shares the goal of understanding the speaker. This communication acts serve to achieve robust team behavior despite adverse conditions, including breaks in communication and other difficulties in achieving the team goals.

Humans also use nonverbal skills such as visual perspective taking and shared attention to establish common ground with others. They orient their own gaze and direct the gaze of their teammate through deictic cues such as pointing gestures in order to establish common ground. Given that visual perspective taking, shared attention, and the use of deictic cues to direct attention are core psychological processes that people use to coordinate joint action about objects and events in the world, robot teammates must be able to display and interpret these behaviors and cues when working with humans in a manner that adheres to human expectations.

Breazeal et al. [58.45] investigated the impact grounding using nonverbal social cues and behavior on task performance by a human–robot team. In a human subject experiment, participants guided Leonardo to perform a physical task using speech and gesture. The robot communicates either implicitly through behavior (such as gaze and facial expressions) or explicitly through nonverbal social cues (i.e., explicit pointing gestures). The robot’s explicit grounding acts include visually attending to the human’s actions to acknowledge their contributions, issuing a short nod to acknowledge the success and completion of the task or subtask, visually attending to the person’s attention directing cues such as to where the human looks or points, looking back to the human once the robot operates on an artifact to make sure its contribution is acknowledged, and pointing to artifacts in the workspace to direct the human’s attention toward them. Both self-reporting via questionnaire and behavioral analysis of video support the hypothesis that implicit nonverbal communication positively impacts human–robot task performance with respect to understandability of the robot, efficiency of task performance, and robustness to errors that arise from miscommunication [58.45].

58.3 Expressive Emotion-Based Interaction

Humans are fundamentally emotional beings. Consequently, human communication and social interaction often includes affective or emotive factors. To support the emotional side of human behavior, researchers are exploring affective interaction and communication between people and robots. To participate in emotion-based interactions, robots must be able to recognize and interpret affective signals from humans, possess their own internal models of emotions (often inspired by psychological theories), and be able to communicate this affective state to others.

The robot's computational model of emotion determines the robot's emotional responses according to its interactions with the external environment and its own internal cognitive-affective state. In psychology, emotional behavior depends upon many internal factors (i. e., the current emotional state, the cognitive state, current desires, physical states, etc.) These physical, cognitive, and affective states are deeply interrelated, and modulate and bias one another.

A growing number of socio-emotional robots have been designed, such as AIBO [58.26], QRIO [58.55], FEELIX [58.56], and many others. In this section, we highlight research in this area using two primary examples: Kismet and the Waseda Emotion Expression Humanoid Robot No. 4 Refined II.

58.3.1 Kismet: Inspiration from Developmental Psychology

Kismet is the first autonomous robot explicitly designed to explore socio-emotive face-to-face interactions with people [58.3]. Research with Kismet focused on exploring the origins of social interaction and communication in people, namely that which occurs between caregiver and infant, though extensive computational modeling guided by insights from psychology and ethology.

Protosocial Responses and Origins of Communication

Early infant-caregiver exchanges are heavily grounded in the regulation of emotion and its expression. Inspired by these interactions, Kismet's cognitive-affective architecture was designed to implement core protosocial responses exhibited by infants given their critical role in normal social development (see Fig. 58.6). Guided by recent psychological theories, Kismet's cognitive-affective architecture (Fig. 58.6) emphasized parallel and interacting systems of emotion and cog-

nition [58.57]. Internally, Kismet's models of emotion interacted intimately with its cognitive systems to influence behavior and goal arbitration [58.58]. Through a process of behavioral homeostasis [58.59], these emotive responses served to restore the robot's internal affective state to a mildly aroused, slightly positive state – corresponding to a state of interest and engagement in people and its surroundings that fosters learning.

One purpose of Kismet's emotive responses was to reflect the degree to which its drives and goals were being successfully met [58.60]. A second purpose was to use emotive communication signals to regulate and negotiate its interactions with people [58.61]. Specifically, Kismet utilized emotive displays to regulate the intensity of playful interactions with people, making sure to keep the complexity of the perceptual stimulus within a range the robot could handle and potentially learn from [58.62]. In effect, Kismet socially negotiated its interaction with people via its emotive responses to have humans help it achieve its goals, and satiate its drives, and maintain a suitable learning environment.

Multimodal Expressive Skills

With respect to its expressive abilities, Kismet generated a wide assortment of facial expressions with corresponding body posture to mirror its affective state. Kismet's facial expressions are generated using an interpolation-based technique over a three-dimensional space (Fig. 58.8). The basis facial postures are designed according to the componential model of facial expressions theorized by *Smith* and *Scott* [58.63] (Fig. 58.9) whereby individual facial features move to convey affective information. The three affective dimensions of the interpolation space correspond to arousal, valence, and stance. These same three attributes are used to affectively appraise the myriad of environmental and internal factors (stimuli, goals, motives, etc.) that contribute to Kismet's *emotional* state [58.58, 64].

In addition to facial expression, Kismet used an expressive vocalization system to generate a wide range of emotive utterances corresponding to joy, sorrow, disgust, fear, anger, surprise, and interest [58.65]. The robot's speech was accompanied with appropriate motor movements of the lips, jaw, and face [58.65].

Breazeal and *Aryananda* [58.66] found that even simple acoustic features (such as pitch mean and energy variance) can be used by the robot to classify the affective prosody of an utterance along valence

Antecedent conditions	Emotio	Behavior	Function
Delay, difficulty in achieving goal of adaptive behavior	Anger, frustration	Complain	Show displeasure to caregiver to modify his/her behavior
Presence of an undesired stimulus	Disgust	Withdraw, reject	Signal rejection or presented stimulus to caregiver
Presence of a threatening, overwhelming stimulus	Fear, distress	Escape	Move away from a potentially dangerous stimuli
Prolonged presence of a desired stimulus	Calm	Engage	Continued interaction with a desired stimulus
Success in achieving goal of active behavior, or praise	Joy	Display pleasure	Reallocate resources to the next relevant behavior (or to reinforce behavior in a learning context)
Prolonged absence of a desired stimulus, or scolding	Sorrow	Display sorrow	Evoke sympathy and attention from caregiver (or to discourage behavior in a learning context)
A sudden, close stimulus	Surprise	Startle response	Alert
Appearance of a desired stimulus	Interest	Orient	Attend to new, salient object
Need of an absent and desired stimulus	Boredom	Seek	Explore environment for desired stimulus

Fig. 58.6 A table summarizing the protosocial emotive responses of Kismet (adapted from *Plutchik* [58.59])

and arousal dimensions. Using these acoustic features and others suggested by *Fernald* [58.67], Kismet could recognize the affective intent in human speech as communicated through vocal prosody that corresponds to praising, soothing, scolding, and attentional bids [58.66].

58.3.2 WE-4RII: A Model of Emotion Inspired by Motion

As another example of a cognitive-affective architecture, the mental model of WE-4RII is shown in Fig. 58.10. In this section, we offer a more technical description of this model to illustrate how such mental models can be computationally implemented [58.68].

The core of this architecture is the emotion model. In dynamics, the movement of objects is described by the equation of motion. The WE-4RII mental model posits that the dynamics of mental transitions might be expressed by similar equations. Hence, the robot implements *equations of emotion* as shown in (58.1) as a second-order differential equation analogous to the equation of motion. Also, three *emotional coefficient matrixes* corresponding to the *emotional inertia*, *emotional viscosity*, and *emotional elasticity matrixes* are introduced:

$$M\ddot{E} + \Gamma\dot{E} + KE = F_{EA}, \quad (58.1)$$

where M is the emotional inertia matrix, Γ is the emotional viscosity matrix, K is the emotional elasticity matrix, and F_{EA} is the emotional appraisal.

Here, *emotional appraisal* stands for the total effect of the stimuli on mental state. By using the *equations*

of emotion, the robot can express the transient aspects of the mental state after the robot senses the stimuli from the environment. Moreover, the robot can express different reactions to the same stimuli by changing the *emotional coefficient matrixes*, and the robot can obtain complex and various mental trajectories, such as a slow reaction or an oscillated reaction [58.69, 70].

Mood Model

Human mental states are affected not only by emotion but also by mood. While emotion is characterized as a strong change that is caused in a short time, moods change more gradually over an extended time. Emotion and mood are closely related, so in this model, moods are represented by introducing the *mood vector* M_d that consists of a *pleasantness axis* and an *activation axis* shown in (58.2)

$$M_d = (M_{dP}, M_{dA}, 0). \quad (58.2)$$

Because the robot's current mental state influences the pleasantness component of the *mood vector* M_{dP} , M_{dP} is defined as the integral of the pleasantness component of the *emotion vector* as shown in (58.3). On the other hand, because the activation component of the *mood vector* M_{dA} is similar to human biological rhythm such as the internal clock [58.69, 70], the van del Pol equation (which is an equation of self-excited oscillation) is applied to describe it as (58.4).

$$M_{dP} = \int E_P dt, \quad (58.3)$$

$$\ddot{M}_{dA} + (1 - M_{dA}^2)\dot{M}_{dA} + M_{dA} = 0. \quad (58.4)$$

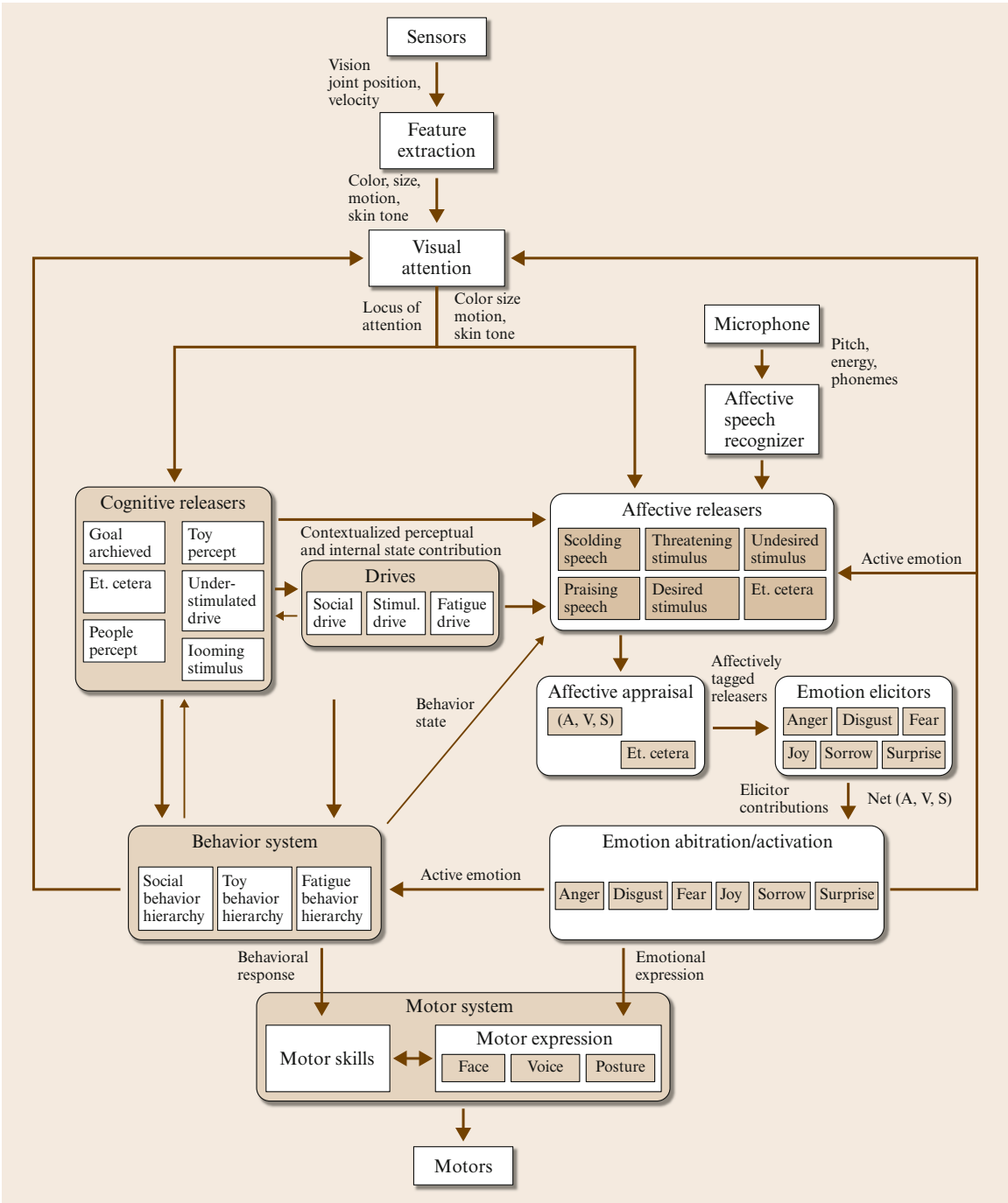


Fig. 58.7 The cognitive-affective architecture of Kismet. The gray boxes with rounded corners represent the cognitive systems responsible for perception, attention, drives, goal arbitration and execution. The white boxes represent the affective processes including affective appraisals of incoming events, basic emotive responses, and expressive motor behavior (vocalizations, facial expressions, etc.)

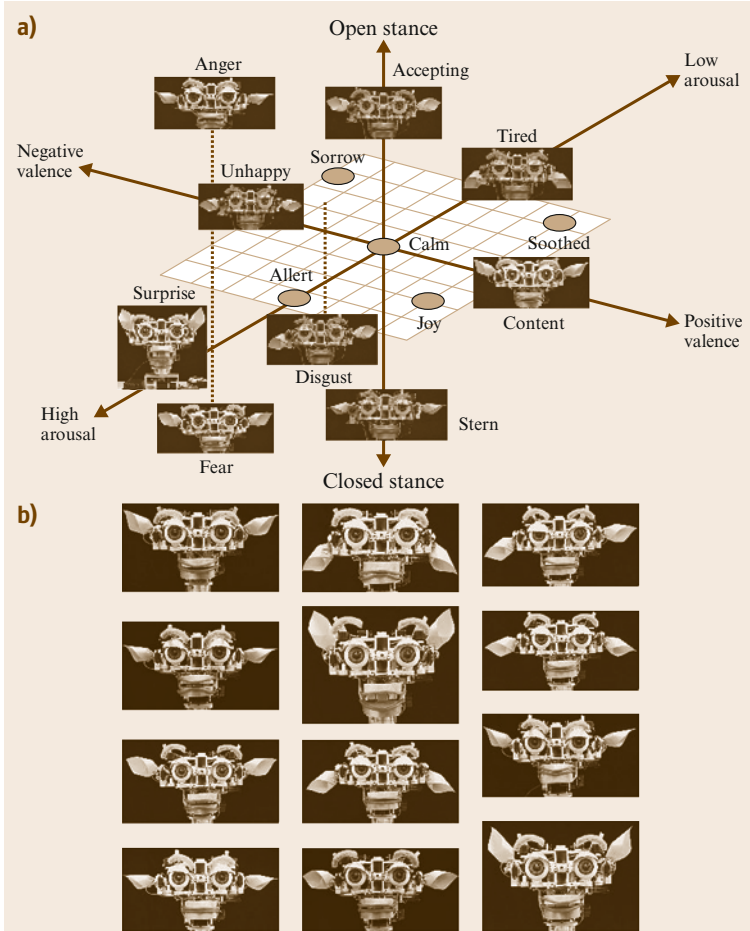


Fig. 58.8 (a) The basis facial poses of Kismet that are interpolated to generate a wide repertoire of expressions. The blend is determined according to the net affective state of the robot (arousal, valence, and stance). (b) A sampling of a range of Kismet's facial expressions from interpolation within this space of basis poses

Need model

For bilateral interaction, humans as well as robots should not only react to an individual's behavior but also behave according to internal needs. As a motivator of spontaneous behavior, a *need model* has been integrated with the mental model. Stimuli from the internal and external environment affect both the needs and the emotions of the robot. Therefore, WE-4RII has a two-layered structure of emotions and needs.

The need state of the robot is described by the *need matrix* N . The *need matrix* of WE-4RII is comprised of three kinds of needs (*appetite*, *need for security*, and *need for exploration*). However, the *need matrix* can be expanded to include other needs. The *need matrixes* N_t at time t and $N_{t+\Delta t}$ at $t + \Delta t$ are described by (58.5).

$$N_{t+\Delta t} = N_t + P_N \Delta N, \quad (58.5)$$

where P_N is the need personality matrix, and ΔN are small differences between two need states.

Equation (58.5) determines the robot need based on the stimuli from the internal and external environments. It is considered a differential equation for a continuous dynamical system. Therefore, (58.5) are named the *equations of need* [58.69, 71].

Facial Expression

WE-4RII can express seven basic emotions defined by Ekman [58.72]. To express them, a three-dimensional mental space consisting of *pleasantness*, *activation*, and *certainty* axes is defined based on psychological research ([58.73, 74]). This is represented as the *emotion vector* E described in (58.6)

$$E = (E_P, E_A, E_C). \quad (58.6)$$

Facial action								
Meaning	Eyebrow frown	Raise brows	Raise upper eyelid	(Raise lower eyelid)	Up turn lip centers	Open mouth	Tighten mouth	Raise chin
Pleasantness	Less	Disgust			More	More	Less	Less
Goal obstacle/discrepancy	More	More						
Anticipated effort	More							
Attentional activity		More	More	(More)				
Certainty		Less				More		
Novelty		More	More					
Personal agency/control		Less	Less			Less		

Fig. 58.9 A mapping of how facial features relate to underlying affective dimensions, inspired by *Smith and Scott* [58.63]. Kismet does not have lower eyelids, so we note this in parentheses

Seven emotions and their corresponding expressions are mapped into the 3-D mental space shown in Fig. 58.11.

WE-4RII determines its emotion by the *emotion vector* *E* passing through each region ([58.69, 70]).

58.4 Socio-cognitive Skills

Socially intelligent robots, however, must understand and interact with animate entities (i. e., people, animals, and other social robots) whose behavior is governed by having a mind and body. In other words, social robots need the ability recognize, understand, and predict human behavior in terms of the underlying mental states such as beliefs, intents, desires, feelings, etc. Psychology calls this ability *theory of mind* (also known as mindreading, mental perception, social common-sense, folk psychology, social understanding, among others).

This section reviews research in implementing models of human socio-cognitive skills and abilities on robots. Social robots will need a diverse repertoire of such skills to realize their full potential in daily human life – to communicate, cooperate, and learn from people in a humancentric and human-compatible manner.

For instance, social robots will need to be aware of people’s goals and intentions so that they can appropriately adjust their behavior to help us as our goals and needs change. They will need to be able to draw their attention flexibly to what we currently find of interest so that their behavior can be coordinated and information can be focused about the same thing. They need to realize that perceiving a given situation from different perspectives impacts what we know and believe to be true about it. This will enable them to bring important

information to our attention that is not easily accessible to us when we need it. Social robots will need to be deeply aware of our emotions, feelings, and attitudes to be able to prioritize what is the most important thing to do for us according to what pleases us or to what we find to be most urgent, relevant, or significant.

Furthermore, the behavior of social robots will need to adhere to people’s expectations. Namely, people will apply their theory of mind to understand the robot in terms of these mental states as well.

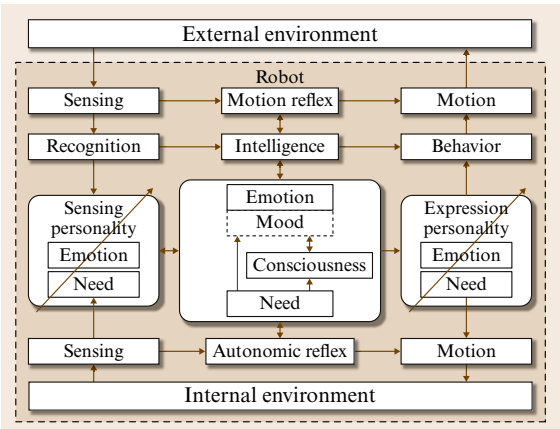


Fig. 58.10 The mental model implemented on the WE-4RII

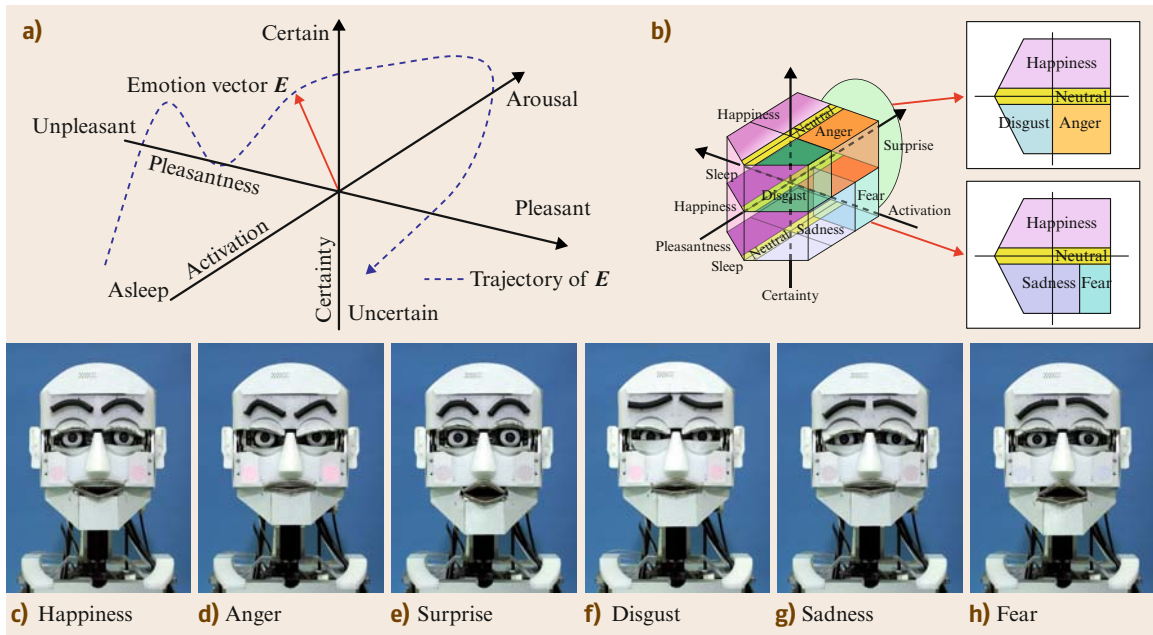


Fig. 58.11a–h The expressions of WF-4RII: (a) 3-D mental space, (b) region mapping of emotion, (c–h) seven basic facial expressions of the WE-4RII

58.4.1 Shared Attention

Scassellati's [58.75] was one of the earliest works to pose the question of how to endow robots with a theory of other minds. Inspired by the theoretical viewpoints proposed from the study of autism (believed to be a deficit of theory of mind), Scassellati implemented a hybrid model of those models proposed by Leslie [58.76] and Baron-Cohen [58.77] where shared attention is viewed to be a critical (and missing) precursor to the theory of mind competence. This hybrid model was implemented on the humanoid robot Cog. The robot was able to exhibit an assortment of social-cognitive skills such as joint attention, distinguishing an entity in the environment as either being animate or inanimate, and imitating only entities deemed to be animate.

Several researchers have explored models of joint reference, guided by insights provided by developmental psychology and autism research [58.78–80]. Normal human infants first demonstrate the ability to share attention with others at 9–12 months of age, such as following the adult's gaze or pointing gestures to the object being referred [58.77, 81]. In these works, joint attention is a learned process. For instance, the robot learns the visual motor mapping from the human's attentional cue (often using head pose as a popular indicator of what

the human is currently looking at) to the motor commands necessary to have the robot look at the same thing. This process is often bootstrapped by having the human look to where the robot initiates its gaze. In Fasel et al. [58.80], the robot learns a model of joint attention because it discovers that the human's gaze is a reliable indicator of where there is something interesting to look.

Thomaz et al. [58.82] explore attention-monitoring behavior of a robot in a social referencing interaction. In the developmental psychology literature, the ability for babies to actively monitor that others are looking at the same thing is a strong indicator of shared attention [58.83]. Social referencing is considered to be an early demonstration of shared attention because the baby looks back and forth between the novel object and the adult's emotive reaction toward that object to learn the association between the two. To implement shared attention, the robot's attentional state is modeled with two related but distinct foci: the current attentional focus (what is being looked at right now) and the referential focus (the current topic of shared focus, i. e., what communication, activities, etc. are about). Furthermore, the robot maintains a model for its own attentional state and a model for the attentional state of the human. The robot uses the heuristic of looking time upon a shared object

to infer the referent of the interaction. Once the referent has been identified, the robot monitors the attention of the human in order to associate their emotional reaction about that object to the intended target.

58.4.2 Emotional Empathy

For humans, the dynamic coupling of like minds through the actions of similar bodies is critical for acquiring human-like intuitions about the internal states of others. *Dautenhahn* [58.2] is one of the earliest works to explore *empathic* mechanisms of understanding others in social robot–robot interaction.

It is likely that emotional empathy in humans is learned, beginning in infancy. Various experiments with human adults have shown a dual affect–body connection whereby posing one’s face into a specific emotive facial expression actually *elicits* the feeling associated with that emotion [58.84, 85]. Hence, imitating the facial expressions of others could cause an infant to feel what the other is feeling, thereby allowing the infant to learn the association of observed emotive expressions of others with the infant’s own internal affective states. Other time-locked multimodal cues

may facilitate learning this mapping, such as affective speech that accompanies emotive facial expressions during social encounters between caregivers and infants.

Using a similar approach, *Breazeal* et al. [58.86] posit that a robot could learn the affective meaning of emotive expressions signaled through another person’s facial expressions, body language, and synchronized multimodal cues such as vocal prosody (see Fig. 58.12). For the robot, certain kinds of stimuli, such as pleasing or soothing tones of speech, have hardwired affective appraisals with respect to arousal and valence [58.66]. This computational model is based on the developmental findings of *Fernald* [58.67] that showed that certain prosodic contours are indicative of different affective intents in infant-directed speech. These affective intents are highly correlated with the corresponding emotive facial expression.

The tasks that couple these heterogeneous processes are face-to-face interactions and imitations. Via dual body–affect pathways, when the robot imitates the emotive facial expressions of others, it evokes the corresponding internal affective state (in terms of arousal and valence variables as described in Sect. 58.3.1) that would ordinarily give rise to the same expression during an emotive response. This is reinforced by affective information coming from the person’s speech signal.

These time-locked multimodal states occur because of the similarity in bodies and body–affect mappings, and they enable the robot to learn to associate its internal affective state with the corresponding observed expression. Thus, the robot uses its own cognitive and affective mechanisms and dual body–affect pathways as a simulator for inferring the human’s affective state as conveyed through behavior. This enables the robot to learn the association between visually observed facial expressions to the underlying affective meaning of those expressions. This is an *empathetic* approach because the robot takes on the corresponding affective state of the human in order to learn and recognize the emotional meaning of the particular facial expression.

58.4.3 Mental Perspective Taking

This section explores this empathetic, self-as-simulator approach further to address more general challenges in endowing robots with mental perspective-taking abilities. These approaches are inspired and informed by theories championed by neuroscience and embodied cognition called *simulation theory*.

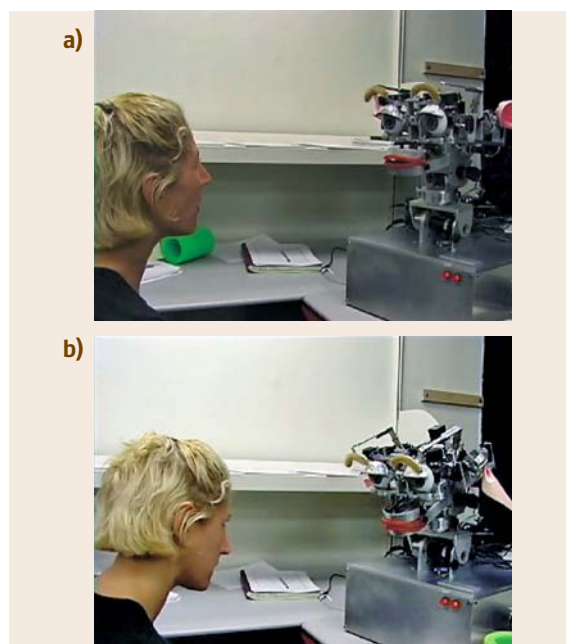


Fig. 58.12a,b Kismet and a young woman mirroring affect. Facial expression and affective tone of voice are tightly correlated: (a) mirroring interest/arousal, (b) mirroring negative affect.

Simulation Theory

Simulation theory holds that certain parts of the brain have dual use; they are used not only to generate our own behavior and mental states, but also simulate the introceptive states of the other person [58.87]. In other words, we engage in a process of perspective taking and mental simulation.

For instance, *Gallese and Goldman* [58.88] proposed that a class of neurons discovered in monkeys (called *mirror neurons*) is a possible neurological mechanism underlying both imitative abilities and simulation theory-type prediction of the behavior of others and their mental states. Further, *Meltzoff and Decety* [58.89] posit that imitation is the critical link in the story that connects the function of mirror neurons to the development of adult mindreading skills. From the field of embodied cognition, *Barsalou et al.* [58.90] present additional evidence from various social embodiment phenomena that, when observing an action, people activate some part of their own representation of that action as well as other cognitive states that relate to that action.

Mirror Systems for Recognizing Actions

Inspired by these theories and findings, *Johnson and Demiris* [58.91] employ a simulation of visual perception to recreate the visual egocentric sensory space and corresponding egocentric behavioral space of the observed agent to increase the accuracy of action recognition. This approach is based on their hierarchical attentive multiple models for execution and recognition (**HAMMER**) architecture that takes an approach inspired by mirror neurons to action recognition and imitation by directly involving the observer's motor system in the action recognition process. Specifically, during observation of another's actions, all of the observer's inverse models (akin to motor programs) are executed in parallel via simulation using forward models, and then compared to the observed action. The one that matches best is selected as being the recognized action. Perceptual perspective taking is needed to provide meaningful data for comparison. The simulated actions used by the observer during recognition must be generated as though from the point of view of the other person. They demonstrate this approach in an experiment where a robot attributes perceptions and recognizes the actions of a second robot [58.91].

Mental Perspective-Taking for Inferring Beliefs and Goals

Gray et al. [58.92] have implemented computational models of simulation-theoretic mechanisms throughout

several systems within Leonardo's cognitive architecture to enable the robot to infer beliefs and goal states of a human collaborator.

The robot reuses its belief-construction systems from the visual perspective of the human to predict the beliefs the human is likely to hold to be true given what he or she can visually observe. This enables the robot to recognize and reason about the beliefs held by a person, even when they diverge from the robot's own beliefs about the same situation.

In psychology, the ability to appreciate the divergent beliefs of another is classically demonstrated by the famous *false-belief* task. In this task, subjects are told a story with pictorial aides that typically proceeds as follows: two children, Sally and Anne, are playing together in a room. Sally places a toy in one of two containers. Sally then leaves the room, and while she is gone, tricky Anne moves the toy into the other container. Sally returns. At this point the human subject is asked "Where will Sally look for the toy?"

The robot, Leonardo, has demonstrated its ability to pass these sorts of false-belief tasks where it observes two humans playing the roles of *Sally* and *Anne* [58.93]. Within the robot's goal-directed behavior system (where schemas relate preconditions and actions with desired outcomes) motor information is used along with perceptual and other contextual clues (such as hierarchically structured task knowledge) to infer the human's goals and how he or she might be trying to achieve them (i. e., plan recognition).

58.4.4 Perspective Taking in Collaboration

By using a simulation-theoretic methodology, mental inferences made across different cognitive systems can interact in interesting and useful ways to support collaborative behavior where a robot offers its human teammate appropriate assistance.

Using Visual Perspective Taking to Resolve Ambiguous Referents

Trafton et al. [58.94] have developed and implemented visual and spatial perspective-taking abilities based on mental simulation to support human-robot interaction and collaboration. Their cognitive architecture, *polyscheme*, is designed to model how humans integrate multiple representational methods, reasoning, and planning methods to keep track of the world, including rich facilities for representing counterfactual worlds. It thus supports simulations of other people's visual perspective to reason about interac-

tions and the world from this alternate point of view.

They have demonstrated these skills in a number of experiments, such as demonstrating the robot's ability to learn how to play *hide and seek* with a person, where the robot learns what makes a good hiding place with respect to being completely occluded from the human seeker's point of view [58.95]. They have also demonstrated the usefulness of this system for a robot that solves a series of perspective-taking problems using the same frames of references and spatial reasoning abilities that astronauts do to facilitate collaborative problem solving – such as repairing a vehicle with another person that has a different vantage point [58.94]. For instance, the robot can handle egocentric requests (i. e., “hand me the cone to my right”), addressee-centric requests (i. e., “hand me the cone to your right”), or object-centered requests (i. e., “hand me the cone in front of the box”).

In Trafton et al. [58.96], a human interacts with the robot using a multimodal interface that supports speech and gesture. The robot's perspective-taking skills are used to resolve ambiguous referents that can arise when a person asks a robot perform an action in relation to an object (i. e., asking the robot to “hand me the wrench” when there are multiple wrenches to choose from). In particular, a visual occlusion in the workspace might hide another candidate wrench from the person's viewpoint but not from the robot's viewpoint (see Fig. 58.13). The robot can infer which is the intended object by taking the visual perspective of the human and applying principles of *joint salience* and *least effort*. If there still remains an ambiguity, the robot can act to resolve it by asking “which one?”

Providing Informational or Instrumental Support

Gray et al. have demonstrated the ability for the Leonardo robot to successfully infer its human partner's beliefs, desires, and intentions from real-time behavior during collaborative tasks. The shared workspace can have either visual occlusions [58.92] or can change dynamically where not all participants know of these changes [58.93]. The robot can integrate these mental state inferences to decide how best to help the person such as offering instrumental support (acting on the environment to help the human complete their goal) or provide informational support (giving relevant information that the person needs to successfully achieve his or her goal).

Consider the following scenario: a helpful robot is introduced to two people, Sally and Anne. All three

watch as Anne hides chips in a box to the left of the robot and cookies in a box to the right. Sally leaves the room, at which point Anne plays a trick on Sally by swapping the contents of the boxes and then locks both boxes with a combination lock. Anne leaves, and Sally soon returns craving the chips she saw placed in one of the boxes. Sally remembers seeing the chips placed in the left box and attempts to open it by working the combination lock. The robot has matching chips and cookies that it can give out. What should the robot do to assist Sally?

Mindreading skills play an important role in this plan recognition scenario where the robot must observe Sally in real time to infer Sally's misconception of where the chips are (Anne switched the location when Sally was out of the room), to infer what her desire is based on her behavior (Sally never explicitly said she wants the chips), and to recognize that Sally's plan for how to get the chips is actually invalid (she is trying to open



Fig. 58.13 Robonaut using visual perspective taking to disambiguate the intended referent when asked to *hand me the wrench*. The human can only see one wrench, but the robot can see both. The robot correctly hands the wrench that both can see

the wrong box). The robot has true knowledge of the situation, and must then reason about how best to help Sally get the object of her desire.

Gray et al. [58.93] combines these three kinds of mental inferences to demonstrate intention recognition with divergent beliefs for collaborative robots. Specifically, for the case of informational support, Leonardo relates its own beliefs about the state of the shared workspace to those of the human based on the visual

perspective of each. If a visual occlusion exists or an event occurs that prevents the human from knowing important information about the workspace, the robot knows to direct the human's attention to bring that information into common ground. For instance, Leonardo points to the box that actually holds the chips. For the case of instrumental support, Leonardo helps the person by directly giving the person a matching bag of chips.

58.5 Conclusion and Further Reading

In this chapter, we have presented some of the principal research trends in social robotics and human–robot interaction. We have relied heavily on examples from our own research to illustrate these trends, and have used excellent examples drawn from other research groups around the world.

From this overview, we have shown that one of the most important goals social robots as applied HRI is the creation of robots that are human-compatible and human-centered in their design. Their differences from human abilities should complement and enhance our strengths. Their similarities to human abilities, such as computationally implementing human cognitive or affective models, may help us to understand ourselves better. We expect that in the coming decades many other researchers, especially young researchers, will actively contribute to the transition from today's robots into capable robot partners of tomorrow.

For further reading, we recommend the following conference proceedings, books, articles, and journal special issues on HRI.

Annual conference proceedings for human–robot interaction:

- Proceedings of the 1st and 2nd Association for Computing Machinery (ACM)/Institute of Electrical and Electronics Engineers (IEEE) International Conference on Human–Robot Interaction (HRI 2006, HRI 2007)
- Proceedings of the 15th IEEE International Symposium on Robot and Human Interactive Communication Getting to Know Socially Intelligent

Robots (Ro-Man, Bellingham 2006) published by the IEEE

- Multidisciplinary Collaboration for Socially Assistive Robotics. Papers from the 2007 Association for the Advancement of Artificial Intelligence (AAAI) Spring Symposium. Technical report SS-07-07, AAAI Press, Menlo Park

Books:

- C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)
- R. W. Picard: *Affective Computing* (MIT Press, Cambridge 1997)
- J.-M. Fellous, M. Arbib: *Who Needs Emotions: The Brain Meets the Robot*. (Oxford Univ. Press, New York 2005)

Review articles:

- T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of social robots., *Robot. Auton. Syst.* **42**, 143 – 166 (2003)

Special issues of journals on HRI:

- R. Murphy, E. Rogers (Eds.): Special Issue on Human–Robot Interaction, *IEEE Trans. Syst. Man Cybernet.* **24**(2) (2004)
- S. Kiesler and P. Hinds (Eds.): Special Issue on Human–Robot Interaction, *Human–Comput. Interact.* **9**(1-2) (2004)
- F. Laschi, C. Breazeal, C. Nakauchi (Eds.): Guest Editorial Special Issue on Human–Robot Interaction, *IEEE Trans. Robot.* **23**(5) (2007)

References

- 58.1 K. Dautenhahn: Getting to know each other – Artificial social intelligence for autonomous robots, *Robot. Auton. Syst.* **16**, 333–356 (1995)
- 58.2 K. Dautenhahn: I could be you: The phenomenological dimension of social understanding, *Cybern. Syst.* **28**, 417–453 (1997)
- 58.3 C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)
- 58.4 H. Miwa, A. Takanishi, H. Takanobu: Experimental study on robot personality for humanoid head robot, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* (2001) pp.1183–1188
- 58.5 T. Tojo, Y. Matsusaka, T. Ishii, T. Kobayashi: A conversational robot utilizing facial and body expressions, *Proc. IEEE SMC2000*, Vol.2 (2000) pp. 858–863
- 58.6 J. Cassell, J. Sullivan, S. Prevost, E. Churchill: *Embodied Conversational Agents* (MIT Press, Boston 2000)
- 58.7 G. Hoffman, C. Breazeal: Robots that work in collaboration with people, *Proc. AAAI Fall Symp. Intersec. Cognit. Sci. Robot.: From Interfaces Intell.* Washington (AAAI, Menlo Park 2004)
- 58.8 K. Dautenhahn, A.H. Bond, L. Canamero, B. Edmonds: *Socially Intelligent Agents: Creating Relationships with Computers and Robots*, Multiagent Syst.Artif. Soc. Simul. Organiz., Vol.3 (Springer, Berlin, Heidelberg 2002)
- 58.9 W.R. Picard: *Affective Computing* (MIT Press, Cambridge 1997)
- 58.10 T. Fong, I. Nourbakhsh, K. Dautenhahn: A survey of social robots, *Robot. Auton. Syst.* **42**, 143–166 (2003)
- 58.11 S. Schaal: Is imitation learning the route to humanoid robots?, *Trends Cognit. Sci.* **3**(6), 233–242 (1999)
- 58.12 J. Solis, K. Chida, K. Suefuji, A. Takanishi: The development of the anthropomorphic flutist robot at Waseda University, *Int. J. Humano. Robots* **3**(2), 1–25 (2006)
- 58.13 Y. Ogura, H. Aikawa, K. Shimomura, H. Kondo, A. Morishima, H. Lim, A. Takanishi: Development of a new humanoid robot WABIAN-2, *Proc. IEEE Int. Conf. Robot. Autom.* (2006) pp. 76–81
- 58.14 H. Miwa, K. Itoh, H. Takanobu, A. Takanishi: Mechanical design and motion control of emotion expression humanoid robot WE-4R, 15th CISM-IFTOMM Symp.Robot Des.Dyn. Cont. ROM04–14 (2004)
- 58.15 F. Tanaka, J.R. Movellan, B. Fortenberry, K. Aisaka: Daily HRI evaluation at a classroom environment: Reports from dance interaction experiments, *Proc. 1st Annu. Conf. Human-Robot Interact. (HRI)* (Salt Lake City 2006) pp. 3–9
- 58.16 H. Lim, S. Hyon, S. Setiawan, S.A. Takanishi: Quasi-human biped walking, *Int. J. Inform. Educ. Res. Robot. Artif. Intell.* **24**(2), 257–268 (2006)
- 58.17 H. Lim, A. Ishii, A. Takanishi: Emotion-based biped walking, *Int. J. Inform. Educ. Res. Robot. Artif. Intell.* **22**(5), 577–586 (2004)
- 58.18 J. Solis, S. Isoda, K. Chida, A. Takanishi, K. Wakamatsu: Anthropomorphic flutist robot for teaching flute playing to beginner students, *Proc. IEEE Int. Conf. Robot. Autom.* (2004) pp.146–151
- 58.19 H. Miwa, K. Itoh, M. Matsumoto, M. Zecca, H. Takanobu, S. Roccella, C.M. Carrozza, P. Dario, A. Takanishi: Effective emotional expressions with emotion expression humanoid robot WE-4RII, *Proc. 2004 IEEE/RSJ Int. Conf. Intell. Robots Syst.* (2004) pp. 2203–2208
- 58.20 S. Roccella, M.C. Carrozza, G. Cappiello, P. Dario, J. Cabibihan, M. Zecca, H. Miwa, K. Itoh, M. Matsumoto, A. Takanishi: Design, fabrication and preliminary results of a novel anthropomorphic hand for humanoid robotics: RCH-1, *Proc. 2004 IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Sendai, 2004) pp. 266–271
- 58.21 H. Miwa, K. Itoh, H. Takanobu, A. Takanishi: Design and control of 9-DoFs emotion expression humanoid arm, 2004 IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp.128–133
- 58.22 F. Iida, M. Tabata, F. Hara: Generating personality character in a Face Robot through interaction with human, *Proc. 7th IEEE Int. Workshop Robot Human Commun.* (1998) pp. 481–486
- 58.23 M. Shimada, T. Minato, S. Itakura, H. Ishiguro: Evaluation of Android Using Unconscious Recognition, *Proc. 6th IEEE-RAS Int. Conf. Humano. Robots* (2006) pp.157–162
- 58.24 K. Berns, J.J. Hirth: Control of facial expressions of the humanoid robot head ROMAN, *Proc. 2006 IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Beijing 2006) pp. 3119–3124
- 58.25 K. Hayashi, Y. Onishi, K. Itoh, H. Miwa, A. Takanishi: Development and evaluation of face robot to express various face shape, *Proc. IEEE Int. Conf. Robot. Autom.* (2006) pp. 481–486
- 58.26 M. Fujita: On activating human communications with pet-type robot AIBO, *Proc. IEEE* **92**(11), 1804–1813 (2004)
- 58.27 K. Wada, T. Shibata, K. Sakamoto, K. Tanie: Long-term interaction between seal robots and elderly people – robot assisted activity at a health service facility for the aged, *Proc. 3rd Int. Symp. Auton. Minirobots Res. Edutain.* (2005) pp. 325–330
- 58.28 C.L. Sidner, C. Lee, C.D. Kidd, N. Lesh, C. Rich: Explorations in engagement for humans and robots, *Artif. Intell.* **166**(1–2), 140–164 (2005)

- 58.29 C. Breazeal, A. Brooks, J. Gray, G. Hoffman, C. Kidd, H.J. Lee Lieberman, A. Lockerd, D. Chilongo: Tute-lage and collaboration for humanoid robots, *Int. J. Humano. Robots* **1**(2), 315–348 (2004)
- 58.30 M. Mori: Bukimi no tani the uncanny valley, *Energy* **7**(4), 33–35 (1970)
- 58.31 W. Stiehl, J. Lieberman, C. Breazeal, L. Basel, L. Lalla, M. Wolf: Design of a therapeutic robotic companion for relational, affective touch, *Proc. 14th IEEE Workshop Robot Human Interact. Commun.* (2005) pp. 408–415
- 58.32 H. Kozima: An anthropologist in the children's world: A field study of children's everyday interaction with an interactive robot, *Proc. Int. Conf. Dev. Learn.* (ICDL) (Bloomington 2006)
- 58.33 R. Gockley, R. Simmons, J. Forlizzi: Modeling affect in socially interactive robots, *Proc. 15th IEEE Int. Symp. Robot Human Interact. Commun.* (2006) pp. 558–563
- 58.34 M.E. Pollack, S. Engberg, J.T. Matthews, S. Thrun, L. Brown, D. Colbry, C. Orosz, B. Peintner, S. Ramakrishnan, J. Dunbar-Jacob, C. McCarthy, M. Montemerlo, J. Pineau, N. Roy: Pearl: A mobile robotic assistant for the elderly, *Proc. AAAI Workshop Autom. Eldercare* (2002)
- 58.35 Y. Fujita: Personal robot PaPeRo, *J. Robot. Mecha-tron.* **14**(1), 60–63 (2002)
- 58.36 M.L. Walters, K. Dautenhahn, S.N. Woods, K.L. Koay: Robotic etiquette: Results from user studies involving a fetch and carry task, *Proc. 2nd ACM/IEEE Int. Conf. Human-Robot Interact.* (Washington 2007) pp. 9–11
- 58.37 R. Gockley, J. Forlizzi, R. Simmons: Natural person-following behavior for social robots, *Proc. Human-Robot Interact.* (2007) pp. 17–24
- 58.38 A.G. Brooks, C.R. Arkin: Behavioral overlays for non-verbal communication expression on a humanoid robot, *Auton. Robots* **22**(1), 55–74 (2007)
- 58.39 H. Fujisawa, K. Shirai: An algorithm for spoken sentence recognition and its application to the speech input-output system, *IEEE Trans. Syst. Man Cybern.* **SMC-4**(5), 475–479 (1974)
- 58.40 T. Kobayashi, Y. Komori, N. Hashimoto, K. Iwata, Y. Fukazawa, J. Yazawa, K. Shirai: Speech conversation system of the musician robot, *Proc. ICAR'85* (1985) pp. 483–488
- 58.41 R. Nisimura, T. Uchida, A. Lee, H. Saruwatari, K. Shikano, Y. Matsumoto: ASKA; receptionist robot with speech dialogue system, *Proc. IEEE Int. Conf. Intell. Syst. Robots* (2002) pp. 1314–1319
- 58.42 H. Kikuchi, M. Yokoyama, K. Hoashi, Y. Hidaki, T. Kobayashi, K. Shirai: Controlling gaze of humanoid in communication with human, *Proc. IROS* (1998) pp. 255–260
- 58.43 S. Hashimoto: Humanoid robots in waseda university: Hadaly2 and WABIAN, *Auton. Robots* **12**(1), 25–38 (2002)
- 58.44 C. Breazeal, A. Edsinger, P. Fitzpatrick, B. Scassel-lati: Active vision systems for sociable robots, *IEEE Trans. Syst. Man Cybern.* **31**(5), 443–453 (2001)
- 58.45 C. Breazeal, C. Kidd, A.L. Thomaz, G. Hoffman, M. Berlin: Effects of nonverbal communication on efficiency and robustness in human-robot team-work, *Proc. Int. Conf. Intell. Robot. Syst.* (2005)
- 58.46 Y. Matsusaka, T. Tojo, T. Kobayashi: Conversation robot participating in group conversation, *Trans. IEICE* **E86-D**(1), 26–36 (2003)
- 58.47 S. Fujie, K. Fukushima, T. Kobayashi: Back-channel feedback generation using linguistic and nonlin-guistic information and its application to spoken dialogue system, *Proc. Interspeech* (2005) pp. 889–892
- 58.48 M. Imai, T. Ono, H. Ishiguro: Physical relation and expression: Joint attention for human- robot in-teraction, *Proc. RO-MAN2001* (2001) pp. 512–517
- 58.49 S. Fujie, R. Miyake, T. Kobayashi: Spoken dialogue system using recognition of user's feedback for rhythmic dialogue, *Proc. Speech Prosody* (2006)
- 58.50 S. Fujie, Y. Ejiri, K. Nakajima, Y. Matsusaka, T. Kobayashi: A conversation robot using head gesture recognition as paralinguistic information, *Proc. IEEE RO-MAN 2004* (2004) pp. 158–164
- 58.51 K. Sakita, K. Ogawara, S. Murakami, K. Kawamura, K. Ikeuchi: Flexible cooperation between human and robot by interpreting human intention from gaze information, *Proc. Int. Conf. Intell. Robot Syst.* (2004) pp. 846–851
- 58.52 A. Brooks, C. Breazeal: Working with robots and objects: revisiting deictic reference for achieving spatial common ground, *Proc. of HRI* (Salt Lake City 2006)
- 58.53 P. Cohen, H. Levesque: Teamwork, *Nous* **25**, 487–512 (1991)
- 58.54 H. Clark, D. Wilkes-Gibbs: Referring as a collabo-rative process, *Cognition* **22**, 1–39 (1986)
- 58.55 F. Tanaka, K. Noda, T. Sawada, M. Fujita: Associated emotion and its expression in an entertainment robot QRIO, *Proc. 3rd Int. Conf. Entertain. Comput.* (Eindhoven 2004) pp. 499–504
- 58.56 L. Canamero: <http://www.felix-growing.org/>, Febru-ary 5 (2007)
- 58.57 C. Breazeal: Function meets style: Insights from emotion theory applied to HRI, *IEEE Trans. Syst. Man Cybern. Part C* **34**(2), 187–194 (2003)
- 58.58 C. Breazeal: Emotion and sociable humanoid robots, *Int. J. Human Comput. Interact.* **58**, 119–155 (2003)
- 58.59 R. Plutchik: Emotions: A general psychoevolution-ary theory,. In: *Approaches to Emotion*, ed. by K. Sherer, P. Elkman (Erlbaum, Hillsdale 1984) pp. 197–219
- 58.60 C. Breazeal, B. Scassellati: Infant-like social inter-actions between a robot and a human caregiver, *Adaptive Behavior* **8**(1), 47–72 (2000)

- 58.61 C. Breazeal: Early experiments using motivations to regulate human-robot interaction, Proc. 1998 Amer. Assoc. Artif. Intell. (AAAI98) Fall Symposium on Emotional and Intelligent: the Tangled knot of Cognition, Techn. Rep., Vol. FS-98-03 (Orlando 1998) pp. 31-36
- 58.62 C. Breazeal, A. Edsinger, P. Fitzpatrick, B. Scasselati: Active vision systems for sociable robots, IEEE Trans. Syst. Man Cybern. **31**(5), 443-453 (2001)
- 58.63 C. Smith, H. Scott: A componential approach to the meaning of facial expressions. In: *The Psychology of Facial Expression*, ed. by J.J. Russell Fernandez-Dols (Cambridge Univ Press, Cambridge 1997) pp. 229-254
- 58.64 C. Breazeal: Regulation and entrainment for human-robot interaction, Int. J. Exp. Robot. **21**(10-11), 883-902 (2003)
- 58.65 C. Breazeal: Emotive qualities in lip synchronized robot speech, Adv. Robot. **17**(2), 97-113 (2003)
- 58.66 C. Breazeal, L. Aryananda: Recognizing affective intent in robot directed speech, Auton. Robots **12**(1), 85-104 (2002)
- 58.67 A. Fernald: Intonation and communicative intent in mother's speech to infants: Is the melody the message?, Child Develop. **60**, 1497-1510 (1989)
- 58.68 K. Itoh, H. Miwa, M. Matsumoto, M. Zecca, H. Takanobu, S. Roccella, M.C. Carrozza, P. Dario, A. Takanishi: Behavior model of humanoid robots based on operant conditioning, Proc. 2005 IEEE-RAS Int. Conf. Humano. Robots (Tsukuba 2005) pp. 220-225
- 58.69 H. Miwa, K. Itoh, H. Takanobu, A. Takanishi: Development of mental model for humanoid robots, 15th CISM-IFTOMM Symp. Robot Des. Dyn. Contr., Vol. ROM04-10 (Montreal 2004)
- 58.70 H. Miwa, T. Okuchi, K. Itoh, H. Takanobu, A. Takanishi: A new mental model for humanoid robots for human friendly communication: Introduction of learning system, mood vector and second order equations of emotion, Proc. IEEE Int. Conf. Robot. Autom. (2003) pp. 3588-3593
- 58.71 H. Miwa, K. Itoh, K. Ito, H. Takanobu, A. Takanishi: Introduction of the need model for humanoid robots to generate active behavior, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (2003) pp. 1400-1406
- 58.72 P. Ekman, W.V. Friesen: The repertoire of nonverbal behavior, Semiotica **1**, 49-98 (1969)
- 58.73 J.A. Russell, M. Bullock: Multidimensional scaling of emotional facial expressions: Similarity from preschool to adults, J. Pers. Social Psychol. **48**, 1290-1298 (1985)
- 58.74 C.A. Smith, P.C. Ellsworth: Patterns of cognitive appraisal in emotion, J. Pers. Social Psychol. **48**, 813-838 (1985)
- 58.75 B. Scasselati: Foundations for a Theory of Mind for a Humanoid Robot. Ph.D. Thesis (MIT, Department of Computer Science and Electrical Engineering 2000)
- 58.76 A. Leslie: How to acquire a representational theory of mind. In: *Metarepresentation: A Multidisciplinary perspective*, ed. by D. Sperber (Oxford Univ Press, Oxford 1994) pp. 197-223
- 58.77 S. Baron-Cohen: Precursors to a theory of mind: Understanding attention in others. In: *Natural Theories of Mind*, ed. by A. Whiten (Blackwell, Oxford 1991) pp. 233-250
- 58.78 B. Scasselati: Mechanisms of shared attention for a humanoid robot, Embodied Cognition and Action: Papers from the 1996 AAAI Fall Symposium (AAAI Menlo Park, 1996)
- 58.79 H. Kozima: Attention-sharing and behavior-sharing in human-robot communication, IEEE Int. Workshop Robot Human Commun. (1998) pp. 9-14
- 58.80 I. Fasel, G.O. Deak, J. Triesch, J. Movellan: Combining embodied models and empirical research for understanding the development of shared attention, Intl. Conf. on Development and Learning (2002) pp. 21-27
- 58.81 G. Butterworth: The ontogeny and phylogeny of joint visual attention. In: *Natural Theories of Mind*, ed. by A. Whiten (Blackwell, Oxford 1991) pp. 223-232
- 58.82 L.A. Thomaz, M. Berlin, C. Breazeal: An embodied computational model of social referencing, Proc. of 14th IEEE Workshop Robot Human Interact. Commun. (Nashville 2005)
- 58.83 S. Feinman: Social referencing in infancy, Merrill-Palmer Quarterly **28**, 445-470 (1982)
- 58.84 F. Strack, L. Martin, S. Stepper: Inhibiting and facilitating conditions of the human smile: A nonobtrusive test of the facial feedback hypothesis, J. Personal. Social Psychol. **54**, 768-777 (1988)
- 58.85 M.P. Niedenthal, W.L. Barsalou, P. Winkielman, S. Krauth-Gruber, F. Ric: Embodiment in attitudes, social perception, and emotion, Personal. Social Psychol. Review **9**(3), 184-211 (2005)
- 58.86 C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, B. Blumberg: Learning from and about others: Towards using imitation to bootstrap the social understanding of others by robots, Artif. Life **11**(1-2), 1-32 (2005), ed by. L. Rocha and F. Almedia e Costa
- 58.87 M. Davies, T. Stone: *Mental Simulation* (Blackwell, Oxford 1995)
- 58.88 V. Gallese, A. Goldman: Mirror neurons and the simulation theory of mind-reading, Trends Cognit. Sci. **2**(12), 493-501 (1998)
- 58.89 A. Meltzoff, J. Decety: What imitation tells us about social cognition: a rapprochement between developmental psychology and cognitive neuroscience, Philos. Trans. R. Soc. Lond. B Biol. Sci. **358**, 491-500 (2003)
- 58.90 L.W. Barsalou, P.M. Niedenthal, A. Barbey, J. Rupert: Social embodiment. In: *The Psychology of Learning and Motivation*, Vol. 43, ed. by B. Ross (Academic Press, San Diego 2003) pp. 43-92

- 58.91 M. Johnson, Y. Demiris: Perceptual perspective taking and action recognition, *Int. J. Adv. Robot. Syst.* **2**(4), 301–308 (2005)
- 58.92 J. Gray, C. Breazeal, M. Berlin, A. Brooks, J. Lieberman: Action parsing and goal inference using self as simulator, *Proc. 14th IEEE Workshop Robot Human Interact. Commun.* (Nashville 2005)
- 58.93 J. Gray, M. Berlin, C. Breazeal: Intention recognition with divergent beliefs for collaborative robots, *Proc. AISB 2007: Symp. Mindful Environ.* (Newcastle upon Tyne 2007)
- 58.94 J.G. Trafton, N. Cassimatis, M. Bugajska, D. Brock, F. Mintz, A. Schultz: Enabling effective human-robot interaction using perspective-taking in robots, *IEEE Trans. Syst. Man Cybern. Part A: Syst. Humans* **35**(4), 460–470 (2005)
- 58.95 G. Trafton, C.A. Shultz, D. Perzanowski, W. Adams, M. Bugajska, N. Cassimatis, D. Brock: Children and robots learning to play hide and seek, *Proc. 1st Annu. Conf. Human-Robot Interact.* (Salt Lake City 2006)
- 58.96 G. Trafton, A. Schultz, M. Bugajska., F. Mintz: Perspective-taking with robots: Experiments and models, *Proc. of 14th IEEE Workshop Robot Human Interact. Commun.* (Nashville 2005)

59. Robot Programming by Demonstration

Aude Billard, Sylvain Calinon, Rüdiger Dillmann, Stefan Schaal

Robot programming by demonstration (PbD) has become a central topic of robotics that spans across general research areas such as human-robot interaction, machine learning, machine vision and motor control.

Robot PbD started about 30 years ago, and has grown importantly during the past decade. The rationale for moving from purely preprogrammed robots to very flexible user-based interfaces for training robots to perform a task is three-fold.

First and foremost, PbD, also referred to as *imitation learning*, is a powerful mechanism for reducing the complexity of search spaces for learning. When observing either good or bad examples, one can reduce the search for a possible solution, by either starting the search from the observed good solution (local optima), or conversely, by eliminating from the search space what is known as a bad solution. Imitation learning is, thus, a powerful tool for enhancing and accelerating learning in both animals and artifacts.

Second, imitation learning offers an implicit means of training a machine, such that explicit and tedious programming of a task by a human user can be minimized or eliminated (Fig. 59.1). Imitation learning is thus a *natural* means of interacting with a machine that would be accessible to lay people.

Third, studying and modeling the coupling of perception and action, which is at the core of imitation learning, helps us to understand the mechanisms by which the self-organization of perception and action could arise during development. The reciprocal interaction of perception and action could explain how competence in motor control can be grounded in rich structure of perceptual variables, and vice versa, how the processes of perception can develop as means to create successful actions.

PbD promises were thus multiple. On the one hand, one hoped that it would make learning faster, in contrast to tedious reinforcement learning methods or trials-and-error learning. On the other hand, one expected that the methods, being user-friendly, would enhance the application of robots in human daily environments. Recent progresses in the field, which we review in this chapter, show that the field has made a leap forward during the past decade toward these goals. In addition, we anticipate that these promises may be fulfilled very soon.

Section 59.1 presents a brief historical overview of robot Programming by Demonstration (PbD), introducing several issues that will be discussed later in this chapter. Section 59.2 reviews engineering approaches to robot PbD with an emphasis on machine learning approaches that provide the robot with the ability to *adapt* the learned skill to different situations (Sect. 59.2.1). This section discusses also the different types of representation that one may use to encode a skill and presents incremental learning techniques to refine the skill progressively (Sect. 59.2.4). Section 59.2.3 emphasizes the importance to give the teacher an active role during learning and presents different ways in which the user can convey cues to the robot to help it to improve its learning. Section 59.2.4 discusses how PbD can be jointly used with other learning strategies to overcome some limitations of PbD. Section 59.3 reviews works that take a more biological approach to robot PbD and develops models of either the cognitive or neural processes of imitation learning in primates. Finally, Sect. 59.4 lists various open issues in robot PbD that have yet been little explored by the field.

59.1 History	1372	59.3 Biologically-Oriented Learning Approaches	1386
59.2 Engineering-Oriented Approaches	1374	59.3.1 Conceptual Models of Imitation Learning.....	1386
59.2.1 Learning a Skill	1374	59.3.2 Neural Models of Imitation Learning.....	1386
59.2.2 Incremental Teaching Methods	1382	59.4 Conclusions and Open Issues in Robot PbD.....	1389
59.2.3 Human-Robot Interaction in PbD	1383	References	1389
59.2.4 Joint Use of Robot PbD with Other Learning Techniques.....	1385		

59.1 History

At the beginning of the 1980s, PbD started attracting attention in the field of manufacturing robotics. PbD appeared as a promising route to automate the tedious manual programming of robots and as way to reduce the costs involved in the development and maintenance of robots in a factory.

As a first approach to PbD, symbolic reasoning was commonly adopted in robotics [59.2–6], with processes referred to as *teach-in*, *guiding* or *play-back* methods. In these works, PbD was performed through manual (teleoperated) control. The position of the end-effector and the forces applied on the object manipulated were stored throughout the demonstrations together with the

positions and orientations of the obstacles and of the target. This sensorimotor information was then segmented into discrete subgoals (keypoints along the trajectory) and into appropriate primitive actions to attain these subgoals (Fig. 59.2). Primitive actions were commonly chosen to be simple point-to-point movements that industrial robots employed at this time. Examples of subgoals would be, e.g., the robot’s gripper orientation and position in relation to the goal [59.4]. Consequently, the demonstrated task was segmented into a sequence of state-action-state transitions.

To take into account the variability of human motion and the noise inherent to the sensors capturing the movements, it appeared necessary to develop a method that would consolidate all demonstrated movements. For this purpose, the state-action-state sequence was converted into symbolic *if-then* rules, describing the states and the actions according to symbolic relationships, such as *in contact*, *close-to*, *move-to*, *grasp-object*, *move-above*, etc. Appropriate numerical definitions of these symbols (i. e., when would an object be considered as *close-to* or *far-from*) were given as prior knowledge to the system. A complete demonstration was thus encoded in a graph-based representation, where each state constituted a graph node and each action a directed link between two nodes. Symbolic reasoning could then unify different graphical representations for the same task by merging and deleting nodes [59.3].

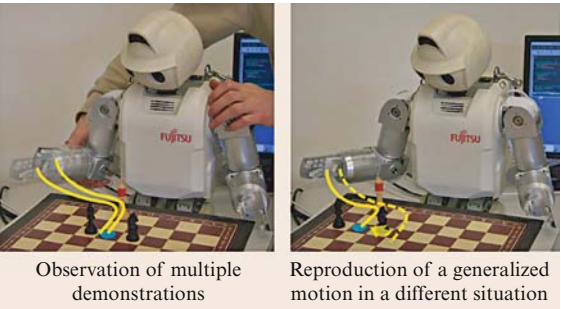


Fig. 59.1 Left: A robot learns how to make a chess move (namely moving the queen forward) by generalizing across different demonstrations of the task performed in slightly different situations (different starting positions of the hand). The robot records its joints’ trajectories and learns to extract *what-to-imitate*, i. e. that the task constraints are reduced to a subpart of the motion located in a plane defined by the three chess pieces. Right: The robot reproduces the skill in a new context (for different initial position of the chess piece) by finding an appropriate controller that satisfies both the task constraints and constraints relative to its body limitation (*how-to-imitate* problem), adapted from [59.1]

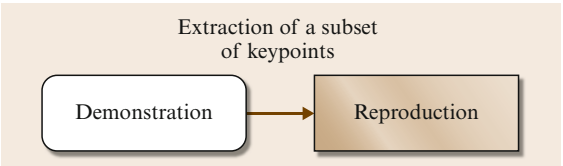


Fig. 59.2 Exact copy of a skill by interpolating between a set of pre-defined keypoints, [59.2]

Muench et al. [59.7] then suggested the use of *machine learning (ML)* techniques to recognize *elementary operators (EOs)*, thus defining a discrete set of basic motor skills, with industrial robotics applications in mind. In this early work, the authors already established several key-issues of **PbD** in robotics. These include questions such as how to generalize a task, how to reproduce a skill in a completely novel situation, how to evaluate a reproduction attempt, and how to better define the role of the user during learning. Muench et al. [59.7] admitted that generalizing over a sequence of discrete actions was only one part of the problem since the controller of the robot also required the learning of continuous trajectories to control the actuators. They proposed that the missing parts of the learning process could be overcome by adapting them to the user who had taken an active role in the teaching process.

These early works highlighted the importance of providing a set of examples that the robot can use:

1. by constraining the demonstrations to modalities that the robot can understand; and
2. by providing a sufficient number of examples to achieve a desired generality.

They noted the importance of providing an adaptive controller to reproduce the task in new situations, that is, how to adjust an already acquired program. The evaluation of a reproduction attempt was also leveraged to the user by letting him/her provide additional examples of the skill in the regions of the learning space that had not been covered yet. Thus, the teacher/expert could control the generalization capabilities of the robot.

In essence, much current works in **PbD** follows a conceptual approach very similar to older work. Recent progresses affected mostly the interfaces at the basis of teaching. Traditional ways of guiding/teleoperating the robot were progressively replaced by more user-friendly interfaces, such as vision [59.9–11], data gloves [59.12], laser range finder [59.13] or kinesthetic teaching (i. e., by manually guiding the robot's arms through the motion) [59.1, 14, 15].

The field progressively moved from simply copying the demonstrated movements to generalizing across sets of demonstrations. Early work adopted a user-guided generalization strategy, in which the robot may ask the user for additional sources of information, when needed. As Machine Learning progressed, **PbD** started incorporating more of those tools to tackle both the perception issue, i. e., how to generalize across demonstrations, and the production issue, i. e., how to generalize the move-

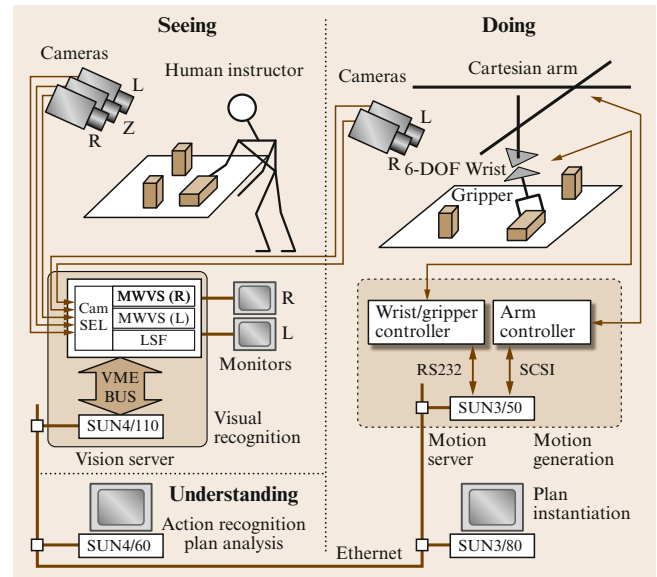


Fig. 59.3 Early approaches to robot programming by demonstration decomposed a task into functional and symbolic units. Temporal dependencies across these units were used to build a hierarchical task plan that drove the robot's reproduction of the task. (After [59.8])

ment to new situations. These tools include *artificial neural networks (ANNs)* [59.16, 17], *radial-basis function networks (RBFs)* [59.18], *fuzzy logic* [59.19], or *hidden markov models (HMMs)* [59.20–24].

As the development of mobile and humanoid robots more animal-like in their behaviors increased, the field went towards adopting an interdisciplinary approach. It took into account evidence of specific neural mechanisms for visuo-motor imitation in primates [59.25, 26] as well as evidence of developmental stages of imitation capacities in children [59.27, 28]. Eventually, the notion of *robot programming by demonstration* was replaced by the more biological labelling of *imitation learning*.

New learning challenges were, thus, set forth. Robots were expected to show a high degree of flexibility and versatility both in their learning system and in their control system in order to be able to interact naturally with human users and demonstrate similar skills (e.g., by moving in the same rooms and manipulating the same tools as humans). Robots were expected more and more to act *human-like* so that their behavior would be more predictable and acceptable.

Thanks to this swift bioinspiration, Robot **PbD** became once again a core topic of research in robotics [59.29–32]. This after the original wave of robotic imitation based on symbolic artificial intelli-

gence methods lost its thrust in the late 1980s. Robot PbD is now a regular topic at the two major conferences on robotics (IROS and ICRA), as well as at

conferences on related fields, such as human-robot interaction (HRI, AISB) and biomimetic robotics (BIOROB, HUMANOIDS).

59.2 Engineering-Oriented Approaches

Engineering-oriented and Machine Learning approaches to robot PbD focus on developing algorithms that are generic in their representation of the skills and in the way they are generated.

Current approaches to represent a skill can be broadly divided between two trends: a low-level representation of the skill, taking the form of a non-linear mapping between sensory and motor information, which we will later refer to as *trajectories encoding*, and, a high-level representation of the skill that decomposes the skill in a sequence of action-perception units, which we will refer to as *symbolic encoding*.

The field has identified a number of key problems that need to be solved to ensure such a generic approach for transferring skills across various agents and situations [59.33, 34]. These have been formulated as a set of generic questions, namely *what to imitate*, *how to*

imitate, *when to imitate* and *who to imitate*. These questions were formulated in response to the large body of diverse work in Robot PbD that could not easily be unified under a small number of coherent operating principles [59.18, 35–39]. The above four questions and their solutions aim at being generic in the sense of making no assumptions on the type of skills that may be transmitted. Who and When to imitate have been largely unexplored so far. Here we essentially review approaches to tackling What and How to imitate, which we refer to as *learning a skill* (what to imitate) and *encoding a skill* (how to imitate). See Fig. 59.4 for an illustration.

59.2.1 Learning a Skill

As mentioned in Sect. 59.1, early approaches to solving the problem of how to generalize a given skill to a new/unseen context consisted in explicitly asking the teacher for further information (Fig. 59.5).

Another way of providing further information to the robot without relying on symbolic/verbal cues consists in doing part of the training in *virtual reality (VR)* or *augmented reality (AR)* and providing the robot with *virtual fixtures*, see Fig. 59.6 and [59.40–43].

Additionally, one sees further methods of learning a skill by allowing a robot to automatically extract the important features characterizing the skill and to look for a controller that optimizes the reproduction of these characteristic features. Determining a *metric of imitation performance* is a key concept at the bottom of these approaches. One must first determine the metric, i. e. de-

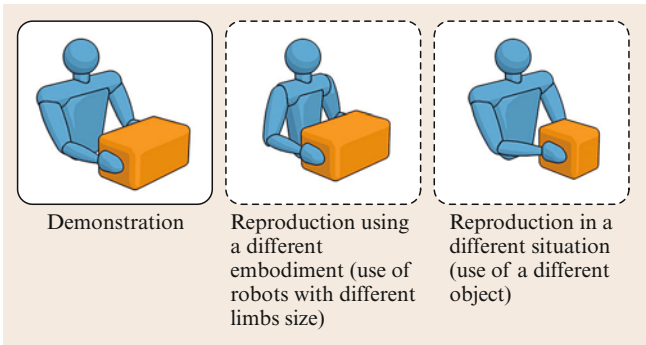


Fig. 59.4 Illustration of the correspondence problems

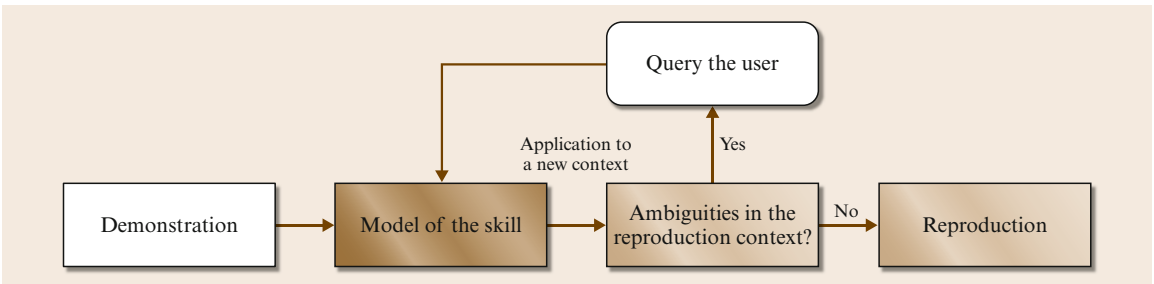


Fig. 59.5 Learning of a skill through a query-based approach, [59.7]

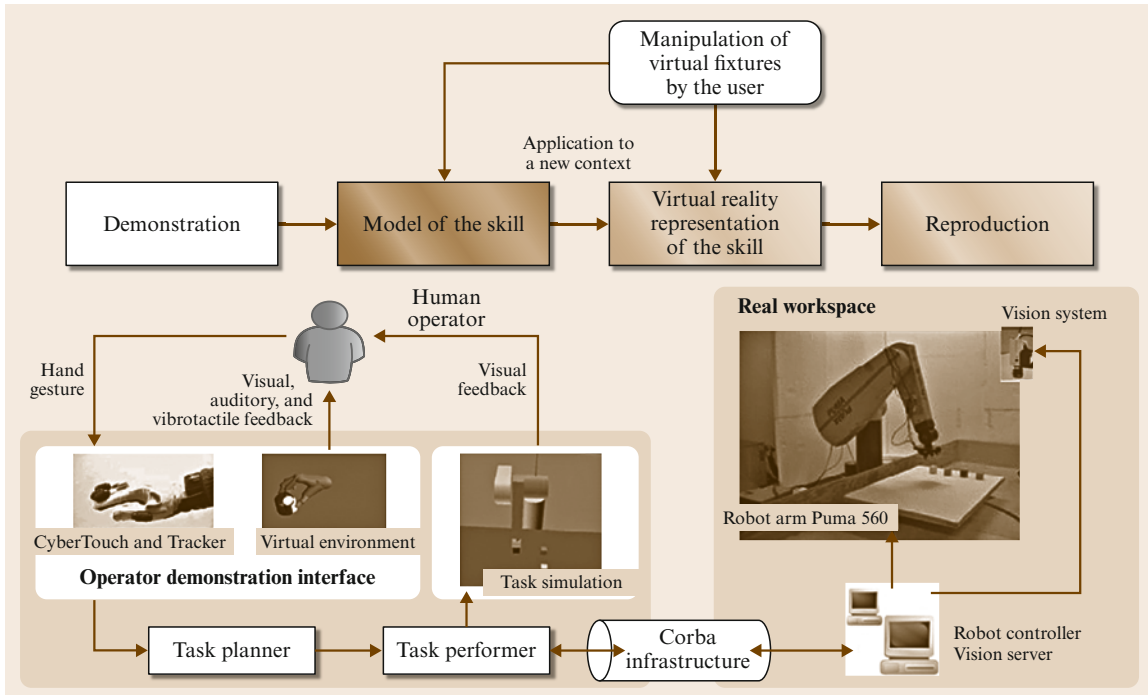


Fig. 59.6 PbD in a virtual reality (VR) setup, providing the robot with virtual fixtures. VR acts as an intermediate layer of interaction to complement real-world demonstration and reproduction. (After [59.44])

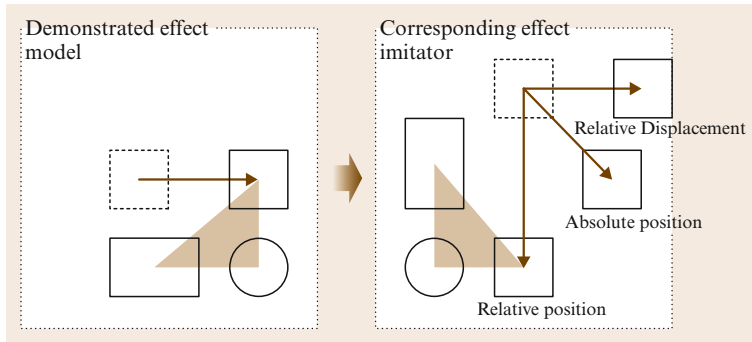


Fig. 59.7 Use of a metric of imitation performance to evaluate a reproduction attempt and find an optimal controller for the reproduction of a task (here, to displace the square in a 2D world). (After [59.45])

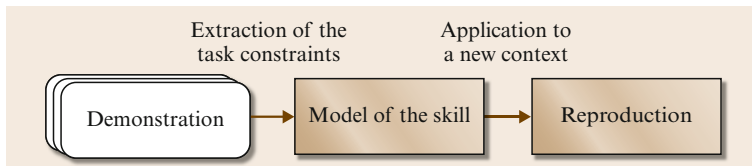


Fig. 59.8 Generalization of a skill by extracting the statistical regularities across multiple observations, [59.46]

termine the weights one must attach to reproducing each of the components of the skill. Once the metric is determined, one can find an optimal controller to imitate by trying to minimize this metric (e.g., by evaluating several reproduction attempts or by deriving the metric

to find an optimum). The metric acts as a cost function for the reproduction of the skill [59.33]. In other terms, a metric of imitation provides a way of expressing quantitatively the user's intentions during the demonstrations and to evaluate the robot's faithfulness at reproducing

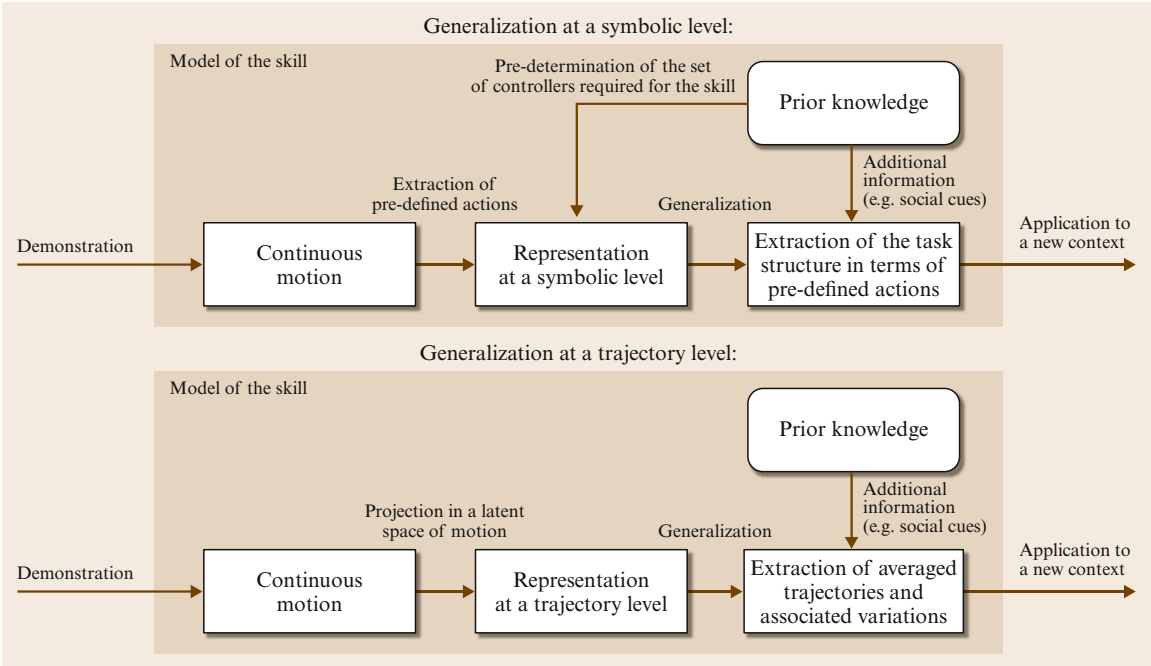


Fig. 59.9 Illustration of the different levels of representation for describing the skill

those. Figure 59.7 shows an illustration of the concept of a metric of imitation performance and its use to drive the robot’s reproduction.

To learn the metric (i.e., to infer the task constraints), one common approach consists in creating a model of the skill based on several demonstrations of the same skill performed in slightly different conditions (Fig. 59.8). This generalization process consists of exploiting the variability inherent to the various demonstrations and to extract the essential components of the task. These essential components should be those that remain unchanged across the various demonstrations [59.1, 46–52].

Figure 59.9 presents a schematic of the learning process by considering either a representation of the skill at a symbolic level or at a trajectory level (these two schemas are detailed versions of the tinted boxes depicted in Fig. 59.8). Table 59.1 summarizes the advantages and drawbacks of the different approaches.

Next, we review a number of specific approaches to learning a skill at the symbolic and trajectory levels.

Symbolic Learning and Encoding of Skills

A large body of work uses a symbolic representation of both the learning and the encoding of skills and tasks [59.7, 22, 49, 50, 52–55]. This symbolic way of

Table 59.1 Advantages and drawbacks of representing a skill at a symbolic/trajectory level

	Span of the generalization process	Advantages	Drawbacks
Symbolic level	Sequential organization of pre-defined motion elements	Allows to learn hierarchy, rules and loops	Requires to pre-define a set of basic controllers for reproduction
Trajectory level	Generalization of movements	Generic representation of motion which allows encoding of very different types of signals/gestures	Does not allow to reproduce complicated high-level skills

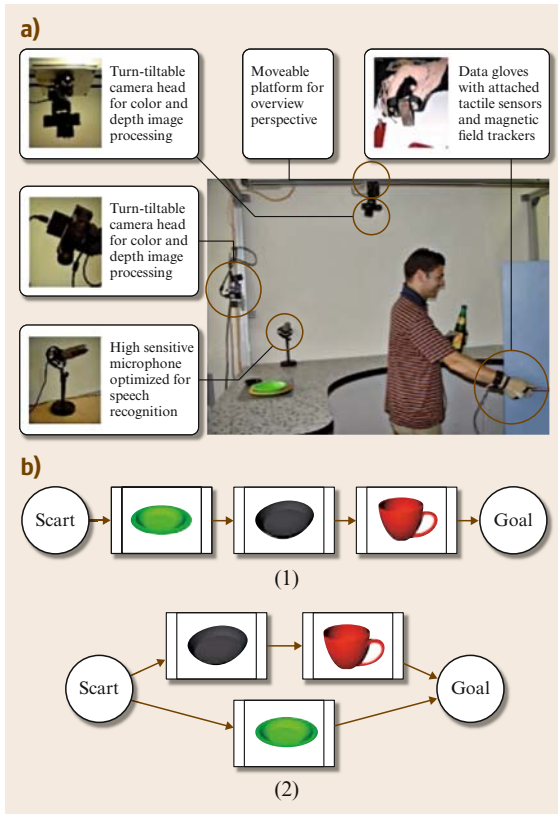


Fig. 59.10 (a) Training center with dedicated sensors. (b) Precedence graphs learned by the system for the ‘setting the table’ task. (a) Initial task precedence graph for the first three demonstrations. (b) Final task precedence graph after observing additional examples. (After [59.50])

encoding skills may take several forms. One common way is to segment and encode the task according to sequences of *predefined* actions, described symbolically. Encoding and regenerating the sequences of these actions can, however, be done using classical machine learning techniques, such as HMM [59.22].

Often, these actions are encoded in a hierarchical manner. In [59.49], a graph-based approach is used to generalize an object’s transporting skill by using a wheeled mobile robot. In the model, each node in the graph represents a complete behaviour and generalization takes place at the level of the topological representation of the graph. The latter is updated incrementally.

References [59.50] and [59.52] follow a similar hierarchical and incremental approach to encode various household tasks (such as setting the table and putting

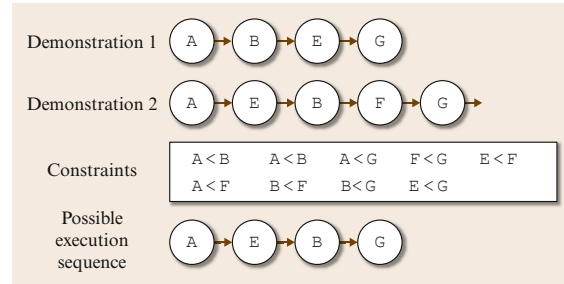


Fig. 59.11 Extraction of the task constraints by considering a symbolic representation of the skill. (After [59.52])

dishes in a dishwasher), Fig. 59.10. There, learning consists in extracting symbolic rules that manages the way each object must be handled (Fig. 59.11).

Reference [59.54] also exploits a hierarchical approach to encoding a skill in terms of pre-defined behaviours. The skill consists in moving through a maze where a wheeled robot must avoid several kinds of obstacles and reach a set of specific subgoals. The novelty of the approach is that it uses a symbolic representation of the skill to explore the teacher’s role in guiding the incremental learning of the robot.

Finally, [59.55] takes a symbolic approach to encoding human motions as sets of pre-defined postures, positions or configuration and considers different levels of granularity for the symbolic representation of the motion. This a priori knowledge is then used to explore the correspondence problem through several simulated setups including motion in joint space of arm links and displacements of objects on a 2D plane (Fig. 59.7).

The main advantage of these symbolic approaches is that high-level skills (consisting of sequences of symbolic cues) can be learned efficiently through an interactive process. However, because of the symbolic nature of their encoding, the methods rely on a large amount of prior knowledge to predefine the important cues and to segment those efficiently (Table 59.1).

Learning and Encoding a Skill at Trajectory-Level

Choosing the variables well to encode a particular movement is crucial, as it already gives part of the solution to the problem of defining what is important to imitate. Work in PbD encodes human movements in either joint space, task space or torque space [59.56–58]. The encoding may be specific to a cyclic motion [59.14], a discrete motion [59.1], or to a combination of both [59.59].

Encoding often encompasses the use of dimensionality reduction techniques that project the recorded signals

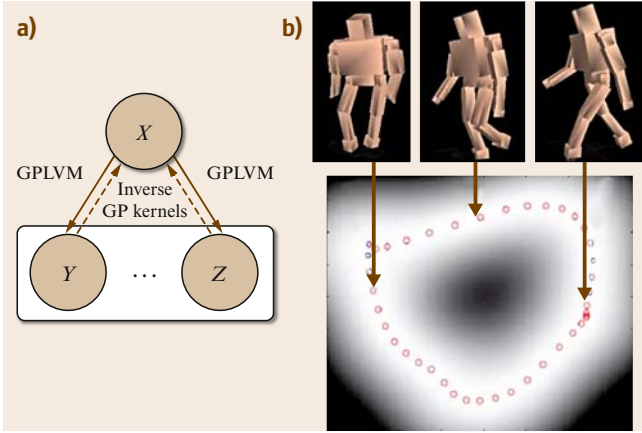


Fig. 59.12a,b Motion learning in a subspace of lower dimensionality by using a non-linear process based on *Gaussian processes (GP)* [59.63]. **(a)** Graphical model of imitation consisting of two pairs of Gaussian Process regression models. In the forward direction, latent variable models map from a low-dimensional latent space X to the human joint space Y and robot joint space Z . In the inverse direction, a regression model maps human motion data to points in the latent space, where two similar human postures in the joint angle space produce two points in the latent space that are close one to the other. Thus, the use of generative regression models allows one to interpolate between known postures in the latent space to create reasonable postures during the reproduction. **(b)** The model provides a smooth certainty estimate in the posture's latent space it infers (*shaded map* from black to white), where training data are represented with circles (here, a walking motion is depicted). Once a latent variable model has been learned, a new motion can be quickly generated from the learned kernels

into a latent space of motion of reduced dimensionality. These techniques may either perform locally linear transformations [59.60–62] or exploit global non-linear methods [59.63–65] (Fig. 59.12).

The most promising approaches to encoding human movements are those that encapsulate the dynamics of the movement into the encoding itself [59.59, 67–70]. Several of these methods are highlighted below.

Skill Encoding Based on Statistical Modeling. One trend of work investigates how statistical learning techniques deal with the high variability inherent to the demonstrations.

For instance, Ude et al. [59.56] use spline smoothing techniques to deal with the uncertainty contained in several motion demonstrations performed in a *joint space* or in a *task space*.

Table 59.2 Probabilistic encoding and reproduction of a skill through *Gaussian mixture regression (GMR)* [59.66]

A dataset $\xi = \{\xi_j\}_{j=1}^N$ is defined by N observations $\xi_j \in \mathbb{R}^D$ of sensory data changing through time (e.g., joint angle trajectories, hand paths), where each datapoint $\xi_j = \{\xi_t, \xi_s\}$ consists of a temporal value $\xi_t \in \mathbb{R}$ and a spatial vector $\xi_s \in \mathbb{R}^{(D-1)}$. The dataset ξ is modelled by a Gaussian mixture model (**GMM**) of K components, defined by the probability density function

$$p(\xi_j) = \sum_{k=1}^K \pi_k \mathcal{N}(\xi_j; \mu_k, \Sigma_k),$$

where π_k are prior probabilities and $\mathcal{N}(\xi_j; \mu_k, \Sigma_k)$ are Gaussian distributions defined by *mean* vectors μ_k and *covariance* matrices Σ_k , whose temporal and spatial components can be represented separately as

$$\mu_k = \{\mu_{t,k}, \mu_{s,k}\}, \quad \Sigma_k = \begin{pmatrix} \Sigma_{tt,k} & \Sigma_{ts,k} \\ \Sigma_{st,k} & \Sigma_{ss,k} \end{pmatrix}.$$

For each component k , the expected distribution of ξ_s given the temporal value ξ_t is defined by

$$\begin{aligned} p(\xi_s | \xi_t, k) &= \mathcal{N}(\xi_s; \hat{\xi}_{s,k}, \hat{\Sigma}_{ss,k}), \\ \hat{\xi}_{s,k} &= \mu_{s,k} + \Sigma_{st,k}(\Sigma_{tt,k})^{-1}(\xi_t - \mu_{t,k}), \\ \hat{\Sigma}_{ss,k} &= \Sigma_{ss,k} - \Sigma_{st,k}(\Sigma_{tt,k})^{-1}\Sigma_{ts,k}. \end{aligned}$$

By considering the complete **GMM**, the expected distribution is defined by

$$p(\xi_s | \xi_t) = \sum_{k=1}^K \beta_k \mathcal{N}(\xi_s; \hat{\xi}_{s,k}, \hat{\Sigma}_{ss,k}),$$

where $\beta_k = p(k | \xi_t)$ is the probability of the component k to be responsible for ξ_t , i.e.,

$$\beta_k = \frac{p(k)p(\xi_t | k)}{\sum_{i=1}^K p(i)p(\xi_t | i)} = \frac{\pi_k \mathcal{N}(\xi_t; \mu_{t,k}, \Sigma_{tt,k})}{\sum_{i=1}^K \pi_i \mathcal{N}(\xi_t; \mu_{t,i}, \Sigma_{tt,i})}.$$

By using the linear transformation properties of Gaussian distributions, an estimation of the conditional expectation of ξ_s given ξ_t is thus defined by $p(\xi_s | \xi_t) \sim \mathcal{N}(\hat{\xi}_s, \hat{\Sigma}_{ss})$, where the parameters of the Gaussian distribution are defined by

$$\hat{\xi}_s = \sum_{k=1}^K \beta_k \hat{\xi}_{s,k}, \quad \hat{\Sigma}_{ss} = \sum_{k=1}^K \beta_k^2 \hat{\Sigma}_{ss,k}.$$

By evaluating $\{\hat{\xi}_s, \hat{\Sigma}_{ss}\}$ at different time steps ξ_t , a generalized form of the motions $\hat{\xi} = \{\xi_t, \hat{\xi}_s\}$ and associated covariance matrices $\hat{\Sigma}_{ss}$ describing the constraints are computed. If multiple constraints are considered (e.g., considering actions $\xi^{(1)}$ and $\xi^{(2)}$ on two different objects), the resulting constraints are computed by first estimating $p(\xi_s | \xi_t) = p(\xi_s^{(1)} | \xi_t) \cdot p(\xi_s^{(2)} | \xi_t)$ and then computing $\mathbb{E}[p(\xi_s | \xi_t)]$ to reproduce the skill. See Fig. 59.14 for an illustration of this method to learning continuous constraints in a set of trajectories. (After [59.66])

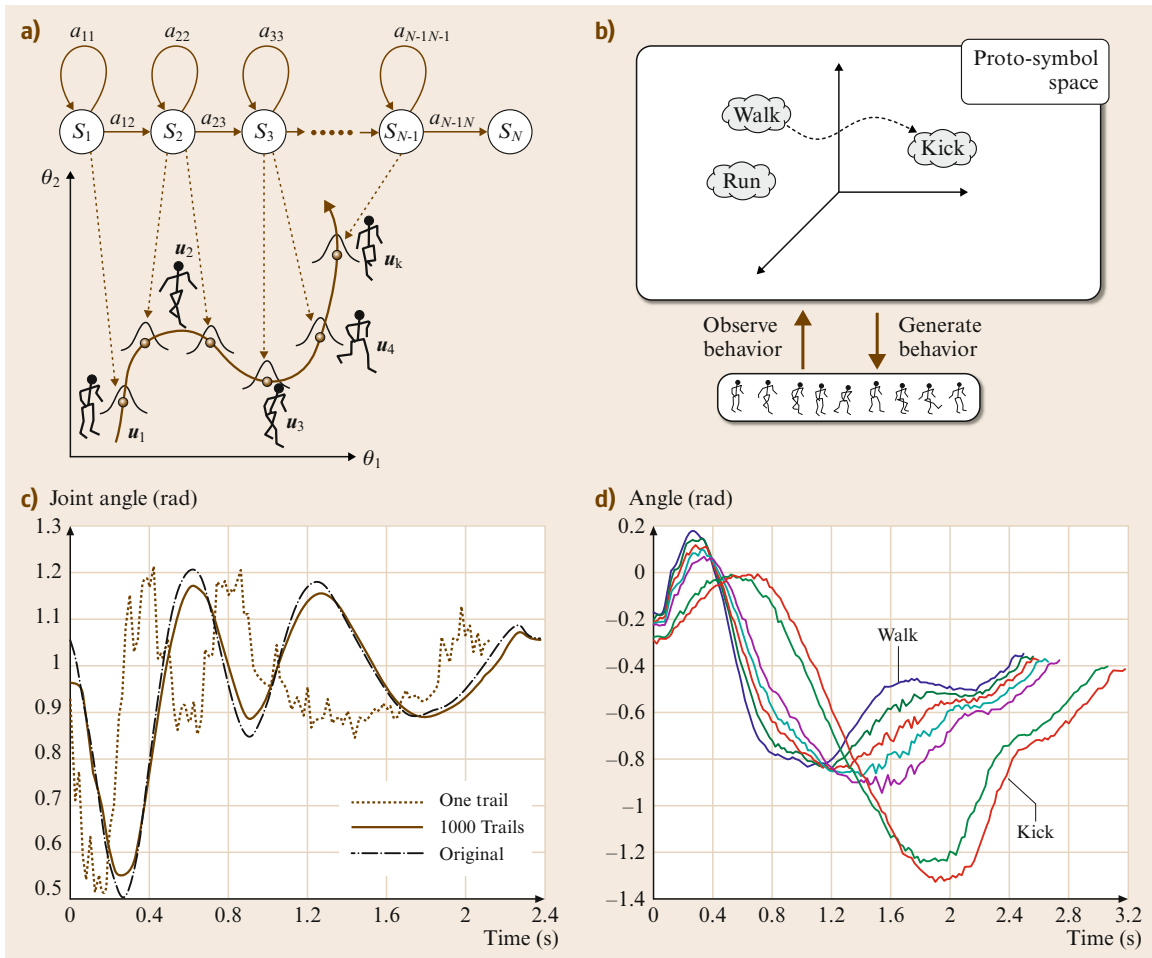


Fig. 59.13a–d Learning of a gesture through the *mimesis* model by using *hidden markov model* (HMM) to encode, recognize and retrieve a generalized version of the motion [59.71]. *Top-left*: Encoding of a full-body motion in a HMM. *Top-right*: Representation of different gestures in a proto-symbol space where the different models are positioned according to the distance between their associated HMM representations. *Bottom-left*: Retrieval of a gesture by using a stochastic generation process based on the HMM representation. *Bottom-right*: Combination of different HMMs to retrieve a gesture combining different motion models

In [59.46], using different demonstrators ensures variability across the demonstrations and quantifies the accuracy required to achieve a *pick and place* task. The different trajectories form a boundary region that is then used to define a range of acceptable trajectories.

In [59.48], the robot acquires a set of sensory variables while demonstrating a manipulation task consisting of arranging different objects. At each time step, the robot stores and computes the mean and variance of the collected variables. The sequence of means and associated variance is then used as a simple generalization

process, providing respectively a generalized trajectory and associated constraints.

A number of authors following such statistically-based learning methods exploited the robustness of *hidden markov models* (HMMs) in order to encode the temporal and spatial variations of complex signals, and to model, recognize and reproduce various types of motions. For instance, Tso et al. [59.23] use HMM to encode and retrieve Cartesian trajectories, where one of the trajectories contained in the training set is also used to reproduce the skill (by keeping the trajectory of the dataset

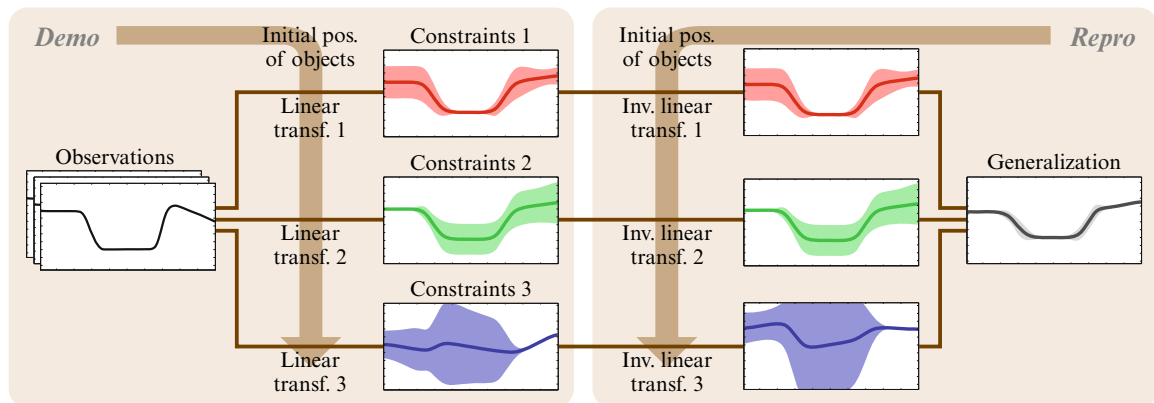


Fig. 59.14 Schematic illustration showing continuous constraints extracted from a set of demonstrations performed in different contexts (namely, different initial positions of objects). Each set of signals recorded during the demonstration is first projected into different latent spaces (through an automatic process of reduction of the dimensionality, such as *principal component analysis* (PCA) or *independent component analysis* (ICA)). Each constraint in this constricted space is then represented probabilistically through *gaussian mixture regression* (GMR) (see Table 59.2). In order to reproduce the task, each constraint is first re-projected in the original data space and a trajectory satisfying optimally all constraints is then computed, after [59.66]

with the highest likelihood, i. e., the one that generalizes the most compared to the others). Yang et al. [59.57] use HMMs to encode the motion of a robot's gripper either in the *joint space* or in the *task space* by considering either the positions or the velocities of the gripper.

The *mimesis model* [59.72–75] follows an approach in which the HMM encodes a set of trajectories, and where multiple HMMs can be used to retrieve new generalized motions through a stochastic process (Fig. 59.13). A drawback of such an approach is that it generates discontinuities in the trajectories regenerated by the system. Interpolation techniques have been proposed to deal with this issue [59.76–78]. Another approach consists of pre-decomposing the trajectories into a set of relevant keypoints and to retrieve a generalized version of the trajectories through spline fitting techniques [59.79–81].

As an alternative to HMM and interpolation techniques, Calinon et al. [59.1] used *gaussian mixture model* (GMM) to encode a set of trajectories, and *gaussian mixture regression* (GMR) to retrieve a smooth generalized version of these trajectories and associated variabilities (Fig. 59.14 and Table 59.2).

Skill Encoding Based on Dynamical Systems. Dynamical systems offer a particularly interesting solution to an imitation process aimed at being robust to perturbations which is robust to dynamical changes in the environment.

The first work to emphasize this approach was that of Ijspeert et al. [59.59], who designed a motor representation based on dynamical systems for encoding movements and for replaying them in various conditions (Fig. 59.15). The approach combines two ingredients: nonlinear dynamical systems for robustly encoding the trajectories, and techniques from non-parametric regression for shaping the attractor landscapes according to the demonstrated trajectories. The essence of the approach is to start with a simple dynamical system, e.g., a set of linear differential equations, and to transform it into a nonlinear system with prescribed attractor dynamics by means of a learnable autonomous forcing term. One can generate both point attractors and limit cycle attractors of almost arbitrary complexity. The point attractors and limit cycle attractors are used to respectively encode discrete (e.g. reaching) and rhythmic movements (e.g. drumming).

Locally weighted regression (LWR) was initially proposed to learn the above system's parameters [59.82–84]. It can be viewed as a memory-based method combining the simplicity of linear least squares regression and the flexibility of nonlinear regression. Further work mainly concentrated on moving on from a memory-based approach to a model-based approach, and moving on from a batch learning process to an incremental learning strategy [59.61, 85, 86]. Schaal et al. [59.86] used *receptive field weighted regression* (RFWR) as a non-parametric approach to incrementally

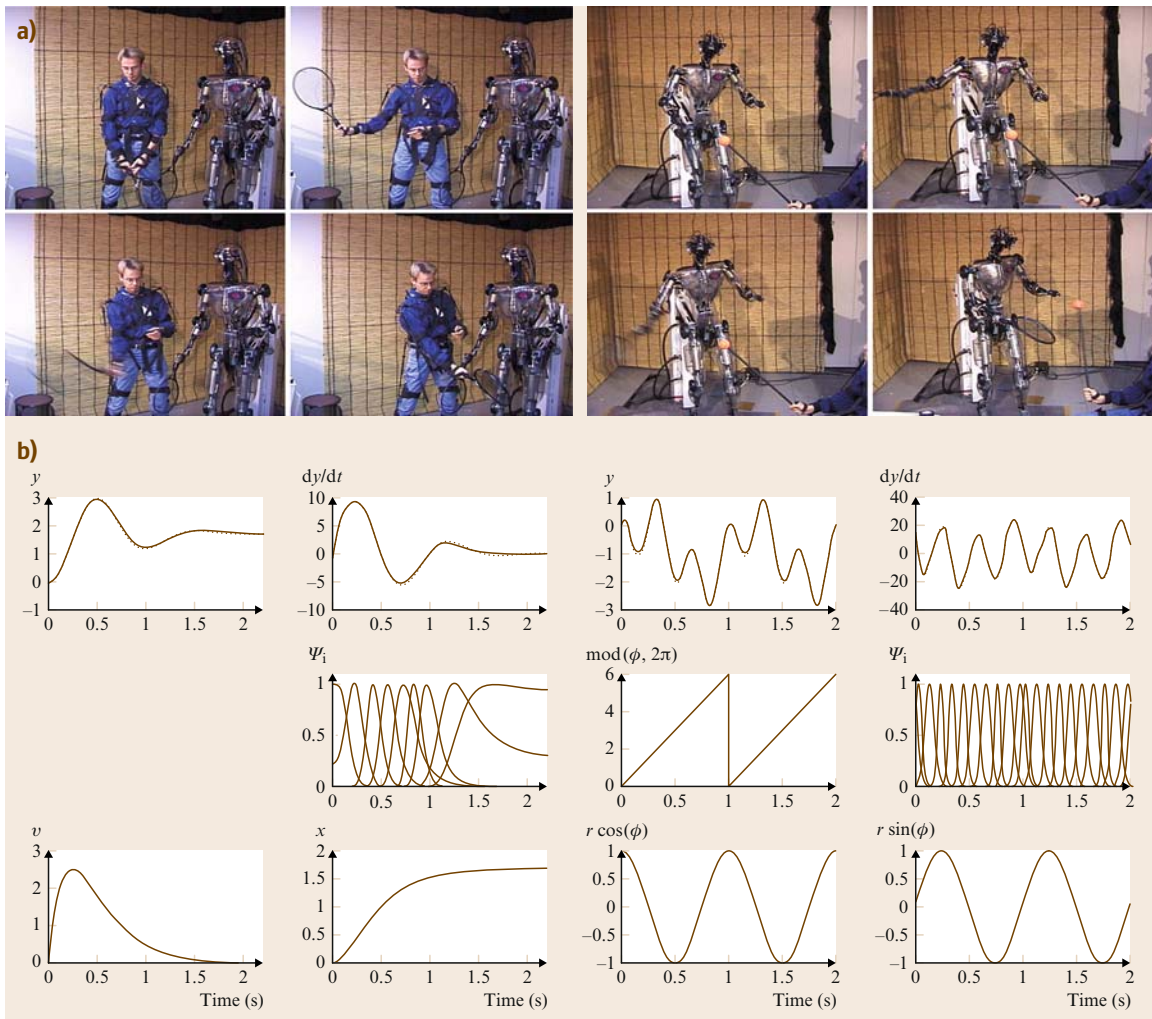


Fig. 59.15 (a) Humanoid robot learning a forehand swing from a human demonstration. (b) Examples of time evolution of the discrete (left) and rhythmic dynamical movement primitives (right). (After [59.59, 88])

learn the fitting function with no need to store the whole training data in memory. Vijayakumar et al. [59.61] then suggested that one uses locally weighted projection regression LWPR to improve the way one approaches operating efficiently in high dimensional space. Hersch et al. [59.87] extended the above dynamical approach to learning combinations of trajectories in a multidimensional space (Fig. 59.16). The dynamical system is modulated by a set of learned trajectories encoded in a gaussian mixture model (Sect. 59.2.1).

The approach offers four interesting properties. First, the learning algorithm, locally weighted regression, is very fast. It does one-shot learning and therefore avoids

the slow convergence exhibited by many neural network algorithms, for instance. Second, the dynamical systems are designed so that, from a single demonstration, they can replay similar trajectories (e.g. tennis swings) with the online modification of a few parameters (e.g. the attractor coordinates). This is of great importance for reusing the dynamical systems in new tasks, i.e. the notion of generalization. Third, dynamical systems are designed to be intrinsically robust in the face of perturbations. Small random noise will not affect the attractor dynamics. In the event of a large constraint (e.g. someone blocking the arm of the robot), feedback terms can be added to the dynamical systems and accordingly mod-

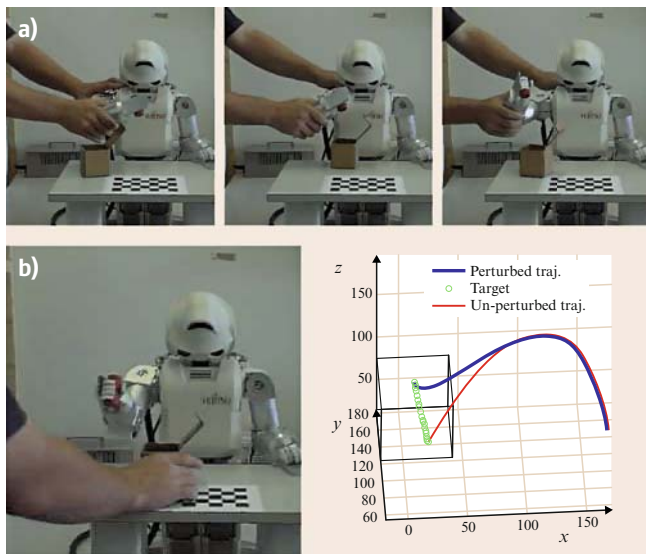


Fig. 59.16a,b Dynamical systems provide a robust robot controller in the face of perturbations during the reproduction of a learned skill. **(a)** The robot is trained through kinesthetic demonstrations by a human trainer (here, the skill consists of putting an object into a box). During these different demonstrations (starting from different positions for both hand and target), the robot records the various velocity profiles its right arm follows. **(b)** Handling of dynamic perturbations during the reproduction. Here, the perturbations are produced by the user who displaces the box during the reproduction attempts. We see that the robot smoothly adapts its original generalized trajectory (*thin line*) to this perturbation (*thick line*). (After [59.87]) ◀

Ito et al. [59.14] proposed another way to encode implicitly the dynamics of bimanual and unimanual tasks, using a recurrent neural networks. The model allows on-line imitation. Of interest is that fact that the network can switch across those motions in a smooth manner when trained separately to encode two different sensorimotor loops (Fig. 59.17). The approach was validated for modelling dancing motions during which the user first initiated imitation and where the roles of the imitator and demonstrator could then be dynamically interchanged. They also considered cyclic manipulation tasks during which the robot continuously moved a ball from one hand to the other and was able to switch dynamically to another cyclic motion that consisted of lifting and releasing the ball.

59.2.2 Incremental Teaching Methods

The statistical approach described previously (Sect. 59.2.1), has its limitations though it is an interesting way to autonomously extract the important features of the task. In addition, it avoids putting too much prior knowledge in the system. For instance, one can expect that requesting multiple demonstrations of a single task would annoy the user. Therefore, **PbD** systems should be capable of learning a task from as few demonstrations as possible. Thus, the robot could start performing its tasks right away and gradually improve its performance

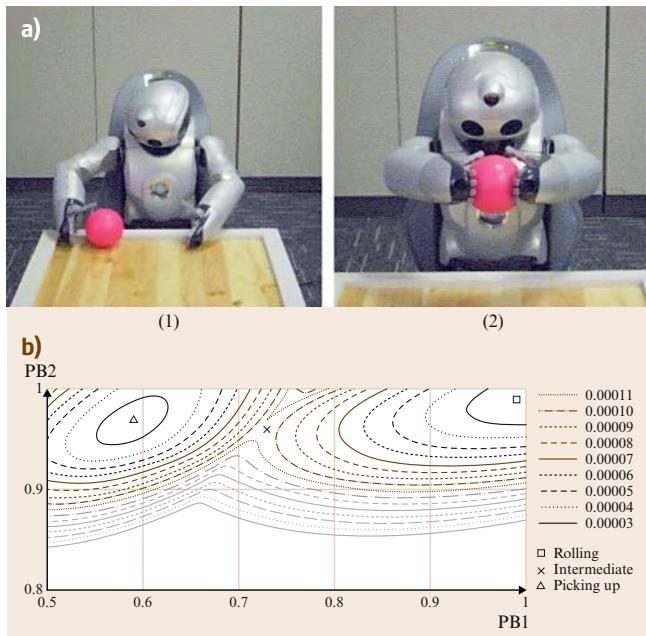


Fig. 59.17a,b Dynamical encoding of the imitation process by using recurrent neural networks. **(a)** Online switching between two interactive behaviours, namely rolling a ball from one hand to the other and lifting the ball. **(b)** Prediction error distribution in the parameters space representing the two behaviours. One can see how the robot can switch smoothly across the behaviors by moving continuously in the parameter space. (After [59.14]) ◀

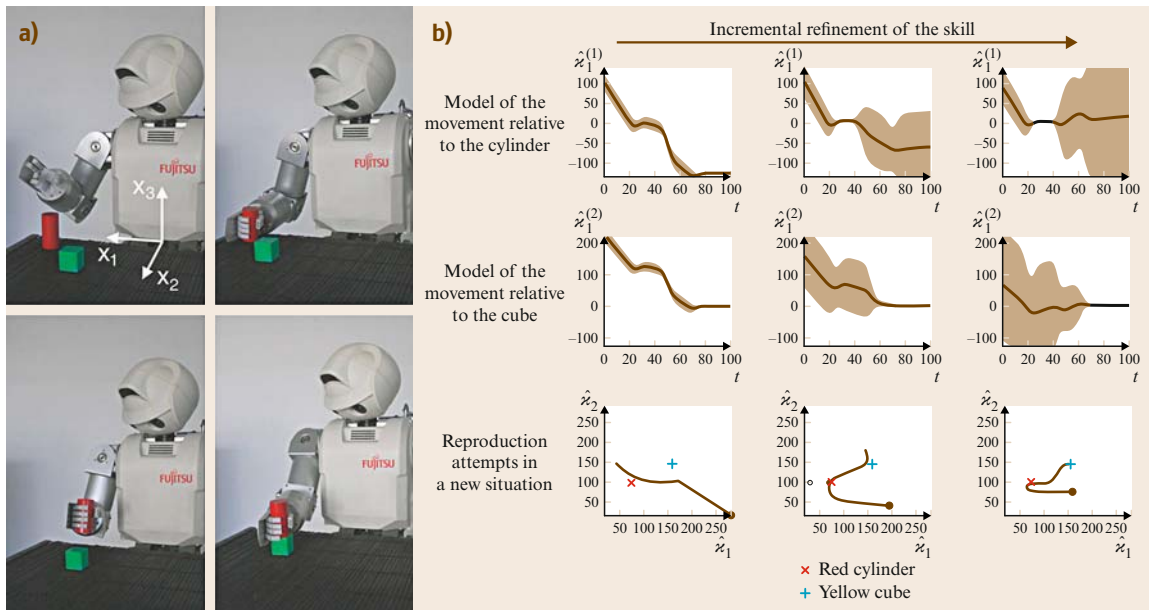


Fig. 59.18a,b Incremental refinement of movements coded in a frame of reference located on the objects that are manipulated. The task constraints are encoded at the trajectory-level. **(a)** Example of a task consisting of grasping a cylinder and placing it on top of a cube. **(b)** Refinement of the *Gaussian mixture regression (GMR)* models representing the constraints all along the movement (see Table 59.2). After a few demonstrations, we see that the trajectories relative to the two objects are highly constrained for particular subparts of the task, namely when reaching for the cylinder (thin envelope around time step 30) and when placing it on top of the cube (thin envelope around time step 100). (After [59.66])

while, at the same time, being monitored by the user. Incremental learning approaches that gradually refine the task knowledge as more examples become available pave the way towards *PbD* systems suitable for real-time human-robot interactions.

Figure 59.18 shows an example of such incremental teaching of a simple skill, namely grasping and placing an object on top of another object, see [59.66] for details.

These incremental learning methods use various forms of deixis, verbal and non-verbal, to guide the robot's attention to the important parts of the demonstration or to particular mistakes it produces during the reproduction of the task. Such incremental and guided learning is often referred to as *scaffolding* or *moulding* of the robot's knowledge. It was deemed most important to allow the robot to learn tasks of increasing complexity [59.54, 89].

Research on the use of incremental learning techniques for robot *PbD* has contributed to the development of methods for learning complex tasks within the household domain from as few demonstrations as possible. Moreover, it contributed to the development and application of machine learning that allow continuous and

incremental refinement of the task model. Such systems have sometimes been referred to as *background knowledge based* or *EM deductive PbD-systems*, as presented in [59.90, 91]. They usually require very few or even only a single user demonstration to generate executable task descriptions.

The main objective of this type of work is to build a meta-representation of the knowledge the robot has acquired on the task and to apply reasoning methods to this knowledge database (Fig. 59.10). This reasoning involves recognizing, learning and representing repetitive tasks.

Pardowitz et al. [59.92] discuss how different forms of knowledge can be balanced in an incrementally learning system. The system relies on building *task precedence graphs*. Task precedence graphs encode hypotheses that the system makes on the sequential structure of a task. Learning the task precedence graphs allows the system to schedule its operations most flexibly while still meeting the goals of the task (see [59.93] for details). Task precedence graphs are directed acyclic graphs that contain a temporal precedence relation that can be learned incrementally. Incremental learning of

task precedence graphs leads to a more general and flexible representation of the task knowledge (Fig. 59.10).

59.2.3 Human-Robot Interaction in PbD

Another perspective adopted by PbD to make the transfer of skill more efficient is to focus on the interaction aspect of the transfer process. As this transfer problem is complex and involves a combination of social mechanisms, several insights from human-robot interaction (HRI) were explored to make efficient use of the teaching capabilities of the human user.

The development of algorithms for detecting *social cues* (given implicitly or explicitly by the teacher during training) and their integration as part of other generic mechanisms for PbD has become the focus of a large body of work in PbD. Such social cues can be viewed as a way to introduce priors in a statistical learning system, and, by so doing, speed up learning. Indeed, several hints can be used to transfer a skill not only by demonstrating the task multiple times but also by highlighting the important components of the skill. This can be achieved by various means and by using different modalities.

A large body of work explored the use of pointing and gazing (Fig. 59.19) as a way to convey the intention of the user [59.41, 95–101]. Vocal deixis, using a standard speech recognition engine, has also been explored widely [59.41, 102]. In [59.50], the user makes vocal comments to highlight the steps of the teaching that are deemed most important. In [59.103, 104], sole the prosody of the speech pattern is looked at, rather than the exact content of the speech, as a way to infer some information on the user's communicative intent.

In [59.94], these social cues are learned through an imitative game, whereby the user imitates the robot. This allows the robot to build a user-specific model of these social pointers, and to become more robust at detecting them.

Finally, a core idea of HRI approach to PbD is that imitation is goal-directed, that is, actions are meant to fulfill a specific purpose and convey the intention of the actor [59.105]. While a longstanding trend in PbD approached the problem from the standpoint of trajectory following [59.106–108] and joint motion replication, see [59.81, 88, 109, 110] and Sect. 59.2.1, recent works, inspired by the above rationale, start from the assump-

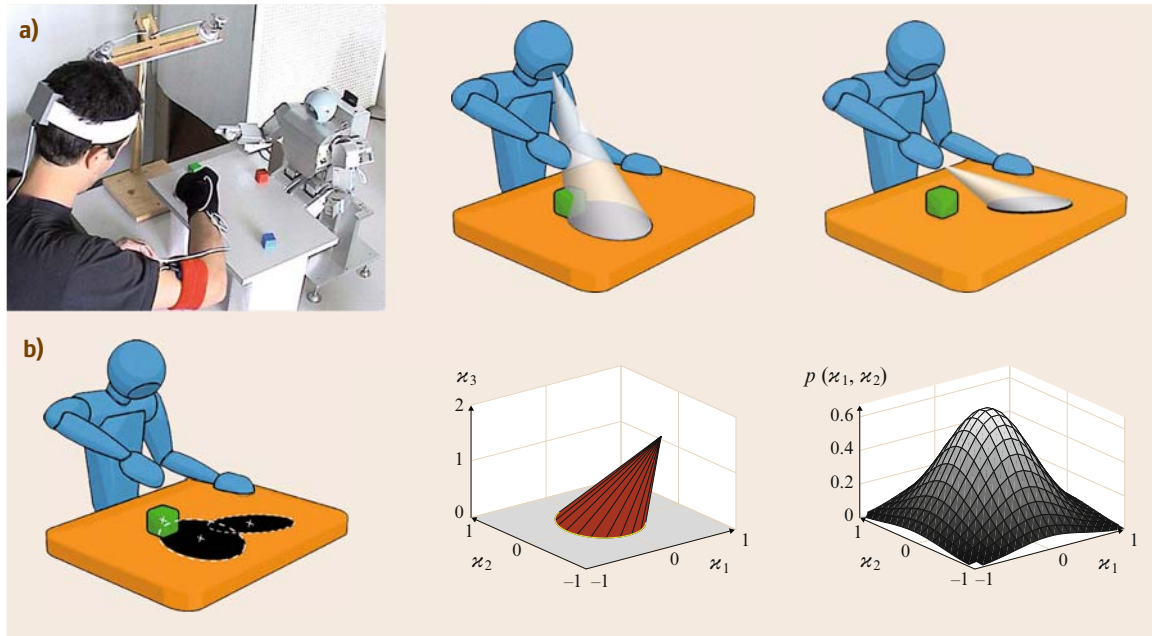


Fig. 59.19a,b Illustration of the use of social cues to speed up the imitation learning process. Here, gazing and pointing towards information are used to select probabilistically the objects relevant for different subparts of a manipulation skill. **(a)** Illustration of the setup proposed in [59.94] to probabilistically highlight the importance of a set of objects through the use of motion sensors recording gazing and pointing directions. **(b)** Probabilistic representation of the intersection between the gazing/pointing cone and the table to estimate where the attention of the user is focused on

tion that imitation is not just about observing and replicating the motion, but rather about *understanding* the goals of a given action. Learning to imitate relies importantly on the imitator's capacity to infer the demonstrator's intentions [59.89, 111]. However, demonstrations may be ambiguous and extracting the intention of the demonstrator requires building a cognitive model of the demonstrator [59.51, 112], as well as exploiting other social cues to provide complementary knowledge [59.54, 89, 113].

Understanding the way humans learn to both extract the goals of a set of observed actions and to give these goals a hierarchy of preference is fundamental to our understanding of the underlying decisional process of imitation. Recent work tackling these issues has followed a probabilistic approach to explain both a goal's derivation and sequential application. The explanation in turn makes it possible to learn manipulatory tasks that require the sequencing of a goal's subsets [59.92, 106, 115, 116].

Understanding the goal of the task is still only half of the picture, as there may be several ways of achieving the goal. Moreover, what is good for the demonstrator may not necessarily be good for the imitator [59.33]. Thus, different models may be allowed to compete to find a solution that is optimal both from the point of view of the imitator and that of the demonstrator [59.77, 117].

59.2.4 Joint Use of Robot PbD with Other Learning Techniques

To recall, a main argument for the development of PbD methods was that it would speed up learning by providing an example of *good solution*. This, however, is

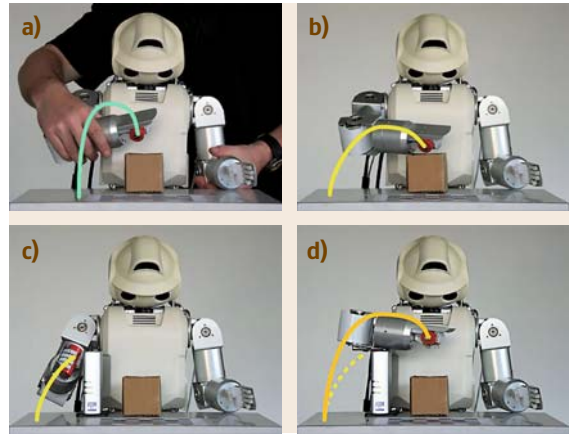


Fig. 59.20a–d Illustration of the use of reinforcement learning to complement PbD. (a) The robot is trained through kinesthetic demonstration on a task that consists of placing a cylinder in a box. (b) The robot reproduces successfully the skill when the new situation is only slightly different from that of the demonstration, using the dynamical system described in Fig. 59.16. (c) The robot fails at reproducing the skill when the context has changed importantly (a large obstacle has been placed in the way). (d) The robot relearns a new trajectory that reproduces the essential aspect of the demonstrated skill, i. e. putting the cylinder in the box, but avoiding the obstacle. (After [59.114])

true insofar that the context for the reproduction is sufficiently similar to that of the demonstration. We have seen in Sect. 59.2 that the use of dynamical systems allows the robot to depart to some extent from a learned trajectory to reach for the target, even when both the object and the hand of the robot have moved

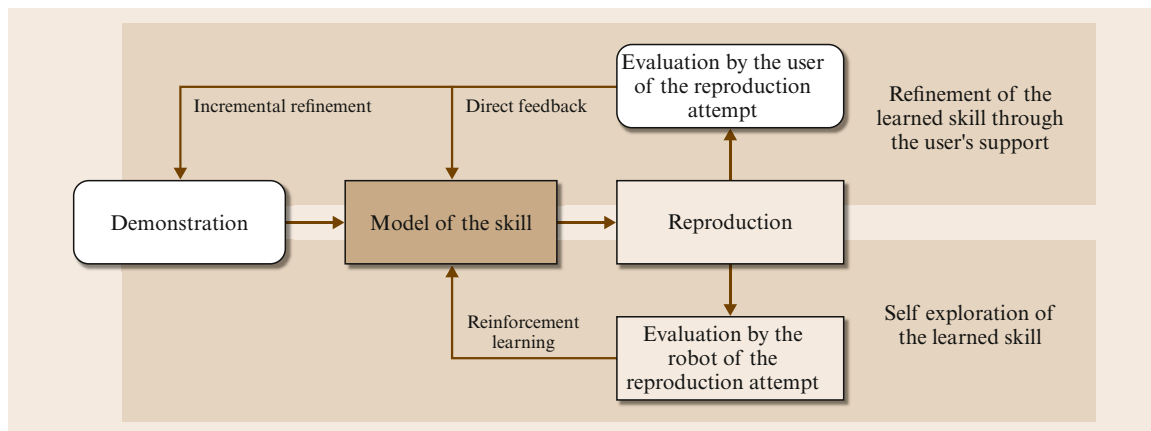


Fig. 59.21 Iterative refinement of the learned skill through teacher's support or through self-exploration by the robot

from the location shown during the demonstration. This approach would not work in some situations, for example, when placing a large obstacle in the robot's path (Fig. 59.20). Besides, robots and humans may differ significantly in their kinematics and dynamics of motion and, although there are varieties of ways to bypass the so-called correspondence problem (Fig. 59.4), re-learning a new model may still be required in special cases.

To allow the robot to learn how to perform a task again in any new situation, it appeared important to combine PbD methods with other motor learning techniques. *Reinforcement learning* (RL) appeared particularly indicated for this type of problem, see Figs. 59.21 and 59.20.

Early work on PbD using RL began in the 1990s and featured learning how to control an inverse pendulum and make it swing up [59.83]. More recent efforts [59.118–120] have focused on the robust control of the upper body of humanoid robots while performing various manipulation tasks.

One can also create a population of agents that copy (mimic) each other so that robots can learn a control strategy by making experiments themselves and by watching others. Such an evolutionary approach, using genetic algorithms, has been investigated by a number of authors, e.g. for learning manipulation skills [59.121], navigation strategies [59.51] or sharing a common vocabulary to name sensoriperception and actions [59.122].

59.3 Biologically-Oriented Learning Approaches

Another important trend in robot PbD takes a more biological stance and develops computational models of imitation learning in animals. We here briefly review recent progresses in this area.

59.3.1 Conceptual Models of Imitation Learning

Bioinspiration is first revealed in the conceptual schematic of the sensorimotor flow which is at the basis of imitation learning that some authors in PbD have followed over the years.

Figure 59.22 sketches the major ingredients of such a conceptual imitation learning system based on sensorimotor representation [59.123]. Visual sensory information needs to be parsed into information about objects and their spatial location in an internal or external coordinate system; the depicted organization is largely inspired by the dorsal (what) and ventral (where) stream as discovered in neuroscientific research [59.124]. As a result, the posture of the teacher and/or the position of the object while moving (if one is involved) should become available. Subsequently, one of the major questions revolves around how such information can be converted into action. For this purpose, Fig. 59.22 alludes to the concept of movement primitives, also called *movement schemas*, *basis behaviors*, *units of action*, or *macro actions* [59.34, 125–127]. Movement primitives are sequences of action that accomplish a complete goal-directed behavior. They could be as simple as an elementary action of an actuator (e.g., *go forward*, *go*

backward, etc.), but, as discussed in [59.123], such low-level representations do not scale well to learning in systems with many degrees-of-freedom. Thus, it is useful for a movement primitive to code complete temporal behaviors, like *grasping a cup*, *walking*, *a tennis serve*, etc.. Figure 59.22 assumes that the perceived action of the teacher is mapped onto a set of existing primitives in an assimilation phase, which is also suggested in [59.128, 129]. This mapping process also needs to resolve the correspondence problem concerning a mismatch between the teacher's body and the student's body [59.130]. Subsequently, one can adjust the most appropriate primitives by learning to improve the performance in an accommodation phase. Figure 59.22 indicates such a process by highlighting the better-matching primitives with increasing line widths. If no existing primitive is a good match for the observed behavior, a new primitive must be generated. After an initial imitation phase, self-improvement, e.g., with the help of a reinforcement-based performance evaluation criterion [59.131], can refine both movement primitives and an assumed stage of motor command generation (see later) until a desired level of motor performance is achieved.

59.3.2 Neural Models of Imitation Learning

Bioinspiration is also revealed in the development of neural network models of the mechanisms at the basis of imitation learning. Current models of Imitation

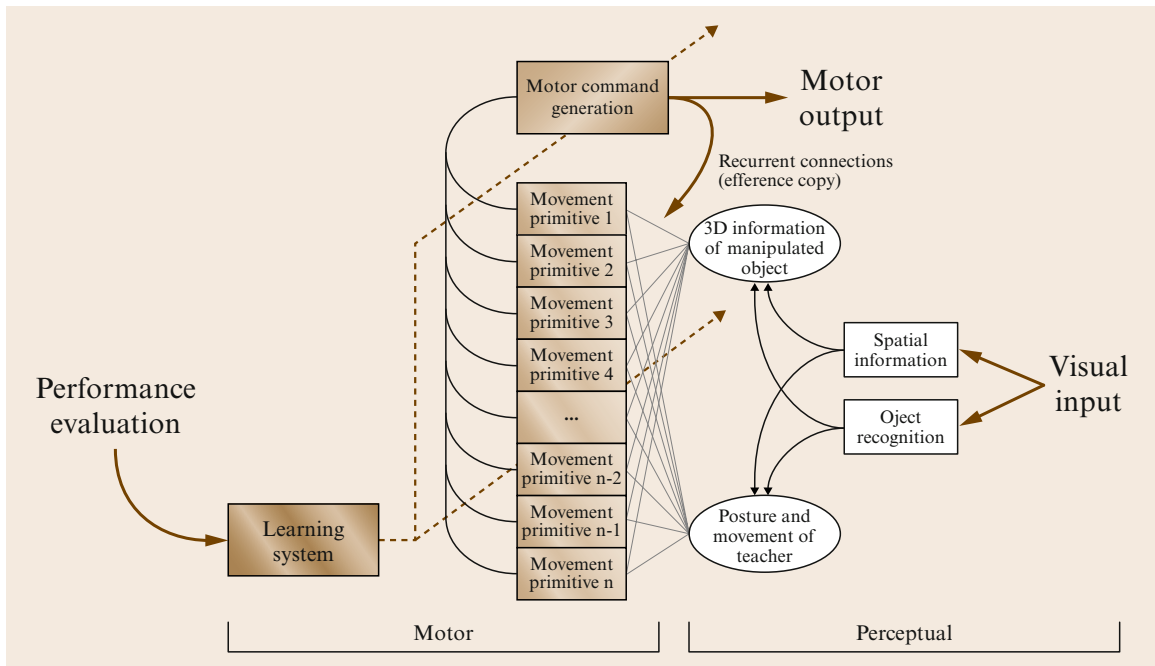


Fig. 59.22 Conceptual sketch of an imitation learning system. The right side of the figure contains primarily perceptual elements and indicates how visual information is transformed into spatial and object information. The left side focuses on motor elements, illustrating how a set of movement primitives competes for a demonstrated behavior. Motor commands are generated from input of the most appropriate primitive. Learning can adjust both movement primitives and the motor command generator. (After [59.123])

Learning all ground their approach on the idea that imitation is at core driven by a *mirror neuron* system. The mirror neuron system refers to a network of brain areas in premotor and parietal cortices that is activated by both the recognition and the production of the same kind of object oriented movements performed by oneself and by others. See [59.26, 132, 133] for recent reports on this system in monkeys and humans, and its link to imitation.

Models of the mirror neuron system assume that somewhere, sensory information about the motion of others and about self-generated motion is coded using the same representation and in a common brain area. The models, however, differ in the way they represent this common center of information. Besides, while all models draw from the evidence of the existence of a mirror neuron circuit and of its application to explain multimodal sensory-motor processing, they, however, go further and tackle the issue of how the brain manages to process the flow of sensorimotor information that is at the basis of how one observes and produces actions. For a comprehensive and critical review of computational

models of the mirror neuron system, the reader may refer to [59.134]. Next, we briefly review the most recent works among those.

One of the first approaches to explaining how the brain processes visuomotor control and imitation was based on the idea that the brain relies on forward models to compare, predict and generate motions [59.129]. These early works set the ground for some of the current neural models of the visuomotor pathway underlying imitation. For instance, recent work by *Demiris* and colleagues [59.117, 128] combines the evidence there is a mirror neuron system which is the basis of recognition and production of basic grasping motions and which proves the existence of forward models for guiding these motions. These models contribute both to the understanding of the mirror neuron system (MNS) in animals as well as its use in controlling robots. For instance, the models successfully reproduce the timing of neural activity in animals while observing various grasping motions and reproducing the kinematics of arm motions during these movements. The models were implemented to control reaching

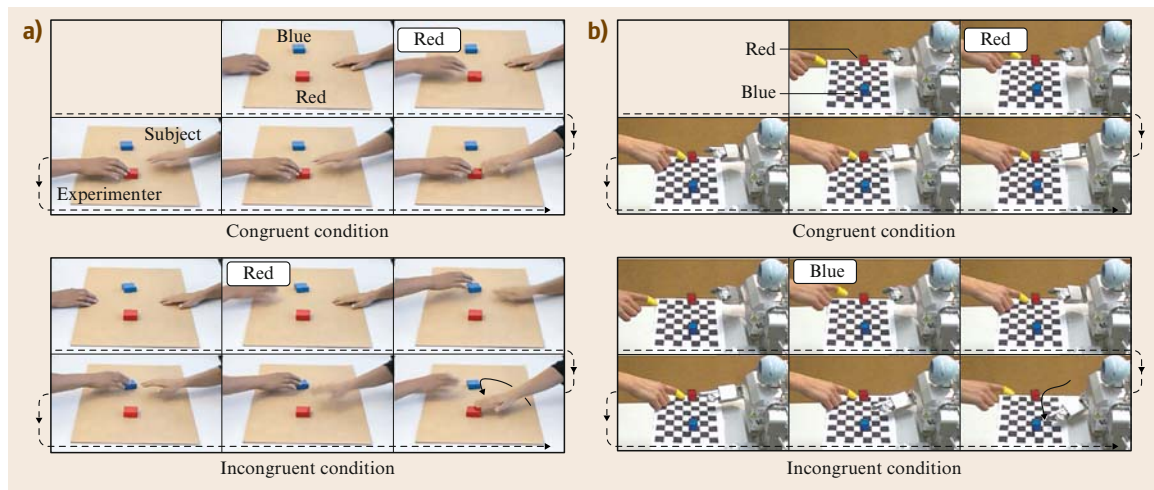


Fig. 59.23 A human experimenter, a human imitator and a robot imitator play a simple imitation game, in which the imitator must point to the object named by the experimenter and not to the object which the experimenter points at. The robot's decision process is controlled by a neural model similar to that found in humans. As a result, it experiences the same associated deficit, known as the principle of *ideomotor compatibility*, stating that *observing the movements of others influences the quality of one's own performance*. When presented with conflicting cues (incongruent condition), e.g. when the experimenter points at a different object than the one named, the robot, like the human subject, either fails to reach for the correct object, or hesitates and later corrects the movement. (After [59.135])

and grasping motions in humanoid and non-humanoid robots.

In contrast, Arbib and colleagues' models take a strong biological stance and follow an evolutionary approach to modeling the mirror neuron system (MNS), emphasizing its application to robotics in a second stage only. They hypothesize that the ability to imitate we find in humans has evolved from Monkeys' ability to reach and grasp and from Chimpanzees' ability to perform simple imitations. They develop models of each of these evolutionary stages, starting from a detailed model of the neural substrate underlying Monkeys' ability to reach and grasp [59.136], extending this model to include the Monkey MNS [59.137] and finally moving to models of the neural circuits underlying a human's ability to imitate [59.138]. The models replicate findings of brain imaging and cell recording studies, as well as make predictions on the time course of the neural activity for motion prediction. As such, the models reconcile a view of forward models of action with an immediate MNS representation of these same actions. They

were used to control reaching and grasping motions in humanoid robotic arms and hands [59.139].

Expanding on the concept of a MNS for sensori-motor coupling, Sauser and Billard [59.140, 141] have explored the use of competitive neural fields to explain the dynamics underlying multimodal representation of sensory information and the way the brain may disambiguate and select across competitive sensory stimuli to proceed to a given motor program. This work explains the principle of *ideomotor compatibility*, by which *observing the movements of others influences the quality of one's own performance*, and develops neural models which account for a set of related behavioral studies [59.142], see Fig. 59.23. The model expands the basic mirror neuron circuit to explain the consecutive stages of sensory-sensory and sensory-motor processing at the basis of this phenomenon. Sauser et al. [59.135] discuss how the capacity for ideomotor facilitation can provide a robot with human-like behavior at the expense of several disadvantages such as hesitation and even mistakes (Fig. 59.23).

59.4 Conclusions and Open Issues in Robot PbD

This chapter aimed at assessing recent progress in modelling the cognitive or neural mechanisms underlying imitation learning in animals and the application of these models to controlling robots, on the one hand. On the other hand, it summarized various machine learning and computational approaches to providing the necessary algorithms for robot programming by demonstration.

Key questions remaining to be assessed by the field are:

- Can imitation use known motor learning techniques or does it require the development of new learning and control policies?
- How does imitation contribute and complement motor learning?
- Does imitation speed up skill learning?
- What are the costs of imitation learning?
- Do models of human kinematics used in gesture recognition drive the reproduction of the task?
- Can one find a level of representation of movement common to both gesture recognition and motor control?
- How could a model extract the intent of the user's actions from watching a demonstration?
- How can we create efficient combinations of imitation learning and reinforcement learning, such that systems can learn in rather few trials?
- What is the role of imitation in human-robot interactions?
- How to find a good balance between providing enough prior knowledge for learning to be fast and incremental, as in task/symbolic learning, and avoid restricting too much the span of learning?

In conclusion, robot PbD contributes to major advances in robot learning and paves the way to the development of robust controllers for both service and personal robots.

References

- 59.1 S. Calinon, F. Guenter, A. Billard: On Learning Representing and Generalizing a Task in a Humanoid Robot, *IEEE Trans. Syst. Man Cybernet.* **37**(2), 286–298 (2007), Special issue on robot learning by observation, demonstration and imitation
- 59.2 T. Lozano-Perez: Robot programming, *Proc. IEEE* **71**(7), 821–841 (1983)
- 59.3 B. Dufay, J.-C. Latombe: An approach to automatic robot programming based on inductive learning, *The Int. J. Robot. Res.* **3**(4), 3–20 (1984)
- 59.4 A. Levas, M. Selfridge: A user-friendly high-level robot teaching system, *Proc. IEEE International Conference on Robotics (1984)* pp. 413–416
- 59.5 A.B. Segre, G. DeJong: Explanation-based manipulator learning Acquisition of planning ability through observation, *IEEE Conference on Robotics and Automation (ICRA) (1985)* pp. 555–560
- 59.6 A.M. Segre: *Machine Learning of Robot Assembly Plans* (Kluwer Academic Publishers, Boston 1988)
- 59.7 S. Muench, J. Kreuziger, M. Kaiser, R. Dillmann: Robot Programming by Demonstration (RPD) – Using Machine Learning and User Interaction Methods for the Development of Easy and Comfortable Robot Programming Systems, *Proc. International Symposium on Industrial Robots (ISIR) (1994)* pp. 685–693
- 59.8 Y. Kuniyoshi, Y. Ohmura, K. Terada, A. Nagakubo, S. Eitoku, T. Yamamoto: Embodied basis of invariant features in execution and perception of whole-body dynamic actionsknacks and focuses of Roll-and-Rise motion, *Robot. Auton. Syst.* **48**(4), 189–201 (2004)
- 59.9 Y. Kuniyoshi, M. Inaba, H. Inoue: Teaching by showing: Generating robot programs by visual observation of human performance, *Proc. International Symposium of Industrial Robots (1989)* pp. 119–126
- 59.10 Y. Kuniyoshi, M. Inaba, H. Inoue: Learning by Watching: Extracting Reusable Task Knowledge from Visual Observation of Human Performance, *IEEE Trans. Robot. Autom.* **10**(6), 799–822 (1994)
- 59.11 S.B. Kang, K. Ikeuchi: A robot system that observes and replicates grasping tasks, *Proc. International Conference on Computer Vision (ICCV) (1995)* pp. 1093–1099
- 59.12 C.P. Tung, A.C. Kak: Automatic learning of assembly task using a DataGlove system, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (1995)* pp. 1–8
- 59.13 K. Ikeuchi, T. Suchiro: Towards an assembly plan from observation, Part I: Assembly task recognition using face-contact relations (polyhedral objects), *Proc. IEEE International Conference on Robotics and Automation (ICRA), Vol. 3 (1992)* pp. 2171–2177
- 59.14 M. Ito, K. Noda, Y. Hoshino, J. Tani: Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model, *Neur. Netw.* **19**(3), 323–337 (2006)

- 59.15 T. Inamura, N. Kojo, M. Inaba: Situation Recognition and Behavior Induction Based on Geometric Symbol Representation of Multimodal Sensorimotor Patterns, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2006) pp. 5147–5152
- 59.16 S. Liu, H. Asada: Teaching and learning of deburring robots using neural networks, Proc. IEEE International Conference on Robotics and Automation (ICRA) (1993) pp. 339–345
- 59.17 A. Billard, G. Hayes: DRAMA, a connectionist architecture for control and learning in autonomous robots, *Adapt. Behav.* **7**(1), 35–64 (1999)
- 59.18 M. Kaiser, R. Dillmann: Building elementary robot skills from human demonstration, Proc. IEEE International Conference on Robotics and Automation (ICRA) (1996) pp. 2700–2705
- 59.19 R. Dillmann, M. Kaiser, A. Ude: Acquisition of elementary robot skills from human demonstration, Proc. International Symposium on Intelligent Robotic Systems (SIRS) (1995) pp. 1–38
- 59.20 J. Yang, Y. Xu, C.S. Chen: Hidden Markov model approach to skill learning and its application in telerobotics, Proc. IEEE International Conference on Robotics and Automation (ICRA) (1993) pp. 396–402
- 59.21 P.K. Pook, D.H. Ballard: Recognizing teleoperated manipulations, Proceedings of IEEE International Conference on Robotics and Automation (ICRA) (1993) pp. 578–585
- 59.22 G.E. Hovland, P. Sikka, B.J. McCarragher: Skill Acquisition from Human Demonstration Using a Hidden Markov Model, Proc. IEEE International Conference on Robotics and Automation (ICRA) (1996) pp. 2706–2711
- 59.23 S.K. Tso, K.P. Liu: Hidden Markov model for intelligent extraction of robot trajectory command from demonstrated trajectories, Proc. IEEE International Conference on Industrial Technology (ICIT) (1996) pp. 294–298
- 59.24 C. Lee, Y. Xu: Online, Interactive Learning of Gestures for Human/Robot Interfaces, Proc. IEEE international Conference on Robotics and Automation (ICRA) (1996) pp. 2982–2987
- 59.25 G. Rizzolatti, L. Fadiga, L. Fogassi, V. Gallese: Resonance behaviors and mirror neurons, *Archives Italiennes de Biologie* **137**(2–3), 85–100 (1999)
- 59.26 J. Decety, T. Chaminade, J. Grezes, A.N. Meltzoff: A PET Exploration of the Neural Mechanisms Involved in Reciprocal Imitation, *Neuroimage* **15**(1), 265–272 (2002)
- 59.27 J. Piaget: *Play, Dreams and Imitation in Childhood* (Norton, New York 1962)
- 59.28 J. Nadel, C. Guerini, A. Peze, C. Rivet: The Evolving Nature of Imitation as a Format for Communication. In: *Imitation in Infancy* (Cambridge University Press, Cambridge 1999) pp. 209–234
- 59.29 M.J. Matarić: Sensory-Motor Primitives as a Basis for Imitation: Linking Perception to Action and Biology to Robotics. In: *Imitation in Animals and Artifacts*, ed. by C. Nehaniv, K. Dautenhahn (MIT Press, Cambridge 2002)
- 59.30 S. Schaal: Nonparametric regression for learning nonlinear transformations. In: *Prerational Intelligence in Strategies, High-Level Processes and Collective Behavior*, ed. by H. Ritter, O. Holland (Kluwer Academic, Dordrecht 1999)
- 59.31 A. Billard: Imitation: a means to enhance learning of a synthetic proto-language in an autonomous robot. In: *Imitation in Animals and Artifacts*, ed. by K. Dautenhahn, C. Nehaniv (MIT Press, Cambridge 2002) pp. 281–311
- 59.32 K. Dautenhahn: Getting to know each other – Artificial social intelligence for autonomous robots, *Robot. Auton. Syst.* **16**(2–4), 333–356 (1995)
- 59.33 C. Nehaniv, K. Dautenhahn: Of Hummingbirds and Helicopters: An Algebraic Framework for Interdisciplinary Studies of Imitation and Its Applications. In: *Interdisciplinary Approaches to Robot Learning*, ed. by J. Demiris, A. Birk (World Scientific Press, Singapore 2000) pp. 136–161
- 59.34 C.L. Nehaniv: Nine Billion Correspondence Problems and Some Methods for Solving Them, Proc. International Symposium on Imitation in Animals and Artifacts (AISB) (2003) pp. 93–95
- 59.35 P. Bakker, Y. Kuniyoshi: Robot See, Robot Do : An Overview of Robot Imitation, Proc. workshop on Learning in Robots and Animals (AISB) (1996) pp. 3–11
- 59.36 M. Ehrenmann, O. Rogalla, R. Zoellner, R. Dillmann: Teaching Service Robots Complex Tasks: Programming by Demonstration for Workshop and Household Environments, Proc. IEEE International Conference on Field and Service Robotics (FRS) (2001)
- 59.37 M. Skubic, R.A. Volz: Acquiring robust, force-based assembly skills from human demonstration, *IEEE Trans. Robot. Autom.* (2000) pp. 772–781
- 59.38 M. Yeasin, S. Chaudhuri: Toward automatic robot programming: learning human skill from visual data, *IEEE Transactions on Systems, Man and Cybernetics, Part B*, **30**(1), 180–185 (2000)
- 59.39 J. Zhang, B. Rössler: Self-Valuing Learning and Generalization with Application in Visually Guided Grasping of Complex Objects, *Robot. Auton. Syst.*, **47**(2–3), 117–127 (2004)
- 59.40 A. Kheddar: Teleoperation based on the hidden robot concept, *IEEE Trans. Syst., Man Cybernet.*, Part A **31**(1), 1–13 (2001)
- 59.41 R. Dillmann: Teaching and Learning of Robot Tasks via Observation of Human Performance, *Robot. Auton. Syst.* **47**(2–3), 109–116 (2004)
- 59.42 J. Aleotti, S. Caselli, M. Reggiani: Leveraging on a Virtual Environment for Robot Programming by Demonstration, *Robot. Auton. Syst.* **47**(2–3), 153–161 (2004)

- 59.43 S. Ekvall, D. Kragic: Grasp Recognition for Programming by Demonstration, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2005) pp. 748–753
- 59.44 J. Aleotti, S. Caselli: Trajectory Clustering and Stochastic Approximation for Robot Programming by Demonstration, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2005) pp. 1029–1034
- 59.45 A. Alissandrakis, C.L. Nehaniv, K. Dautenhahn, J. Saunders: Evaluation of Robot Imitation Attempts: Comparison of the System's and the Human's Perspectives, Proc. ACM SIGCHI/SIGART conference on Human-robot interaction (HRI) (2006) pp. 134–141
- 59.46 N. Delson, H. West: Robot Programming by Human Demonstration: Adaptation and Inconsistency in Constrained Motion, Proc. IEEE International Conference on Robotics and Automation (ICRA) (1996) pp. 30–36
- 59.47 T. Sato, Y. Genda, H. Kubotera, T. Mori, T. Harada: Robot imitation of human motion based on qualitative description from multiple measurement of human and environmental data, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2003) pp. 2377–2384
- 59.48 K. Ogawara, J. Takamatsu, H. Kimura, K. Ikeuchi: Extraction of essential interactions through multiple observations of human demonstrations, IEEE Trans. Indust. Electron. **50**(4), 667–675 (2003)
- 59.49 M.N. Nicolescu, M.J. Mataric: Natural Methods for Robot Task Learning: Instructive Demonstrations, Generalization and Practice, Proc. International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS) (2003) pp. 241–248
- 59.50 M. Pardowitz, R. Zoellner, S. Knoop, R. Dillmann: Incremental Learning of Tasks from User Demonstrations, Past Experiences and Vocal Comments., IEEE Trans. Syst., Man Cybernet. **37**(2), 322–332 (2007), Special issue on robot learning by observation, demonstration and imitation
- 59.51 B. Jansen, T. Belpaeme: A computational model of intention reading in imitation, Robot. Auton. Syst. **54**(5), 394–402 (2006)
- 59.52 S. Ekvall, D. Kragic: Learning Task Models from Multiple Human Demonstrations, Proc. IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN) (2006) pp. 358–363
- 59.53 H. Friedrich, S. Muench, R. Dillmann, S. Bocionek, M. Sassin: Robot programming by Demonstration (RPD): Supporting the induction by human interaction, Mach. Learn. **23**(2), 163–189 (1996)
- 59.54 J. Saunders, C.L. Nehaniv, K. Dautenhahn: Teaching robots by moulding behavior and scaffolding the environment, Proc. ACM SIGCHI/SIGART conference on Human-Robot Interaction (HRI) (2006) pp. 118–125
- 59.55 A. Alissandrakis, C.L. Nehaniv, K. Dautenhahn: Correspondence Mapping Induced State and Action Metrics for Robotic Imitation, IEEE Trans. Syst. Man Cybernet. **37**(2), 299–307 (2007), Special issue on robot learning by observation, demonstration and imitation
- 59.56 A. Ude: Trajectory Generation from Noisy Positions of Object Features for Teaching Robot Paths, Robot. Auton. Syst. **11**(2), 113–127 (1993)
- 59.57 J. Yang, Y. Xu, C.S. Chen: Human Action Learning via Hidden Markov Model, IEEE Trans. Syst. Man Cybernet. **27**(1), 34–44 (1997)
- 59.58 K. Yamane, Y. Nakamura: Dynamics Filter – Concept and implementation of online motion Generator for human figures, IEEE Trans. Robot. Autom. **19**(3), 421–432 (2003)
- 59.59 A.J. Ijspeert, J. Nakanishi, S. Schaal: Learning Control Policies For Movement Imitation and Movement recognition, Neural Inform. Process. Syst. (NIPS) **15**, 1547–1554 (2003)
- 59.60 S. Vijayakumar, S. Schaal: Locally Weighted Projection Regression: An On Algorithm for Incremental Real Time Learning in High Dimensional Spaces, Proc. International Conference on Machine Learning (ICML) (2000) pp. 288–293
- 59.61 S. Vijayakumar, A. D'souza, S. Schaal: Incremental Online Learning in High Dimensions, Neur. Comput. **17**(12), 2602–2634 (2005)
- 59.62 N. Kambhatla: Local Models and Gaussian Mixture Models for Statistical Data Processing. Ph.D. Thesis (Oregon Graduate Institute of Science and Technology, Portland 1996)
- 59.63 A. Shon, K. Grochow, R. Rao: Robotic Imitation from Human Motion Capture using Gaussian Processes, Proc. IEEE/IRAS International Conference on Humanoid Robots (Humanoids) (2005)
- 59.64 K. Grochow, S.L. Martin, A. Hertzmann, Z. Popovic: Style-based inverse kinematics, Proc. ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH) (2004) pp. 522–531
- 59.65 K.F. MacDorman, R. Chalodhorn, M. Asada: Periodic nonlinear principal component neural networks for humanoid motion segmentation, generalization, and generation, Proc. International Conference on Pattern Recognition (ICPR), Vol. 4 (2004) pp. 537–540
- 59.66 S. Calinon, A. Billard: What is the Teacher's Role in Robot Programming by Demonstration? – Toward Benchmarks for Improved Learning, Interact. Stud. **8**(3), 441–464 (2007), Special Issue on Psychological Benchmarks in Human-Robot Interaction
- 59.67 D. Bullock, S. Grossberg: VITE and FLETE: neural modules for trajectory formation and postural control. In: *Volitional control*, ed. by W.A. Hersberger (Elsevier, Amsterdam 1989) pp. 253–297
- 59.68 F.A. Mussa-Ivaldi: Nonlinear force fields: a distributed system of control primitives for represent-

- ing and learning movements, IEEE International Symposium on Computational Intelligence in Robotics and Automation (1997) pp. 84–90
- 59.69 P. Li, R. Horowitz: Passive velocity field control of mechanical manipulators, IEEE Trans. Robot. Autom. **15**(4), 751–763 (1999)
- 59.70 G. Schoener, C. Santos: Control of movement time and sequential action through attractor dynamics: a simulation study demonstrating object interception and coordination, Proc. International Symposium on Intelligent Robotic Systems (SIRS) (2001)
- 59.71 T. Inamura, H. Tanie, Y. Nakamura: Keyframe Compression and Decompression for Time Series Data based on Continuous Hidden Markov Models, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2003) pp. 1487–1492
- 59.72 T. Inamura, I. Toshima, Y. Nakamura: Acquiring Motion Elements for Bidirectional Computation of Motion Recognition and Generation. In: *Experimental Robotics VIII*, Vol. 5, ed. by B. Siciliano, P. Dario (Springer, Berlin Heidelberg 2003) pp. 372–381
- 59.73 T. Inamura, N. Kojo, T. Sonoda, K. Sakamoto, K. Okada, M. Inaba: Intent Imitation using Wearable Motion Capturing System with On-line Teaching of Task Attention, Proc. IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2005) pp. 469–474
- 59.74 D. Lee, Y. Nakamura: Stochastic Model of Imitating a New Observed Motion Based on the Acquired Motion Primitives, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2006) pp. 4994–5000
- 59.75 D. Lee, Y. Nakamura: Mimesis Scheme using a Monocular Vision System on a Humanoid Robot, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2007) pp. 2162–2168
- 59.76 S. Calinon, A. Billard: Learning of Gestures by Imitation in a Humanoid Robot. In: *Imitation and Social Learning in Robots, Humans and Animals: Behavioural, Social and Communicative Dimensions*, ed. by K. Dautenhahn, C.L. Nehaniv (Cambridge Univ. Press, Cambridge 2007) pp. 153–177
- 59.77 A. Billard, S. Calinon, F. Guenter: Discriminative and Adaptive Imitation in Uni-Manual and Bi-Manual Tasks, Robot. Auton. Syst. **54**(5), 370–384 (2006)
- 59.78 S. Calinon, A. Billard: Recognition and Reproduction of Gestures using a Probabilistic Framework combining PCA, ICA and HMM, Proc. International Conference on Machine Learning (ICML) (2005) pp. 105–112
- 59.79 S. Calinon, F. Guenter, A. Billard: Goal-Directed Imitation in a Humanoid Robot, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2005) pp. 299–304
- 59.80 T. Asfour, F. Gyrfas, P. Azad, R. Dillmann: Imitation Learning of Dual-Arm Manipulation Tasks in Humanoid Robots, Proc. IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2006) pp. 40–47
- 59.81 J. Aleotti, S. Caselli: Robust trajectory learning and approximation for robot programming by demonstration, Robot. Auton. Syst. **54**(5), 409–413 (2006)
- 59.82 C.G. Atkeson: Using Local Models to Control Movement, Adv. Neur. Inform. Process. Syst. (NIPS) (1990) pp. 316–323
- 59.83 C.G. Atkeson, A.W. Moore, S. Schaal: Locally Weighted Learning for Control, Artif. Intell. Rev. **11**(1–5), 75–113 (1997)
- 59.84 A.W. Moore: Fast, Robust Adaptive Control by Learning only Forward Models. In: *Adv. Neur. Inform. Process. Syst. (NIPS)*, Vol. 4, ed. by S. Editor (Morgan Kaufmann, San Francisco 1992)
- 59.85 S. Schaal, C.G. Atkeson: From Isolation to Cooperation: An Alternative View of a System of Experts. In: *Adv. Neur. Inform. Process. Syst. (NIPS)*, Vol. 8, ed. by S. Editor (Morgan Kaufmann, San Francisco 1996) pp. 605–611
- 59.86 S. Schaal, C.G. Atkeson: Constructive Incremental Learning from Only Local Information, Neur. Comput. **10**(8), 2047–2084 (1998)
- 59.87 M. Hersch, F. Guenter, S. Calinon, A.G. Billard: Learning Dynamical System Modulation for Constrained Reaching Tasks, Proc. IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2006) pp. 444–449
- 59.88 A.J. Ijspeert, J. Nakanishi, S. Schaal: Movement imitation with nonlinear dynamical systems in humanoid robots, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2002) pp. 1398–1403
- 59.89 C. Breazeal, M. Berlin, A. Brooks, J. Gray, A.L. Thomaz: Using perspective taking to learn from ambiguous demonstrations, Robot. Auton. Syst. **54**(5), 385–393 (2006)
- 59.90 Y. Sato, K. Bernardin, H. Kimura, K. Ikeuchi: Task analysis based on observing hands and objects by vision, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2002) pp. 1208–1213
- 59.91 R. Zoellner, M. Pardowitz, S. Knoop, R. Dillmann: Towards Cognitive Robots: Building Hierarchical Task Representations of Manipulations from Human Demonstration, Proc. IEEE International Conference on Robotics and Automation (ICRA) (2005) pp. 1535–1540
- 59.92 M. Pardowitz, R. Zoellner, R. Dillmann: Incremental learning of task sequences with information-theoretic metrics, Proc. European Robotics Symposium (EUROS) (2005)
- 59.93 M. Pardowitz, R. Zoellner, R. Dillmann: Learning sequential constraints of tasks from user demonstrations, Proc. IEEE-RAS International Conference

- on Humanoid Robots (Humanoids) (2005) pp. 424–429
- 59.94 S. Calinon, A. Billard: Teaching a Humanoid Robot to Recognize and Reproduce Social Cues, Proc. IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN) (2006) pp. 346–351
- 59.95 B. Scassellati: Imitation and Mechanisms of Joint Attention: A Developmental Structure for Building Social Skills on a Humanoid Robot, Lect. Notes Comput. Sci. **1562**, 176–195 (1999)
- 59.96 H. Kozima, H. Yano: A robot that learns to communicate with human caregivers, Proc. International Workshop on Epigenetic Robotics (2001)
- 59.97 H. Ishiguro, T. Ono, M. Imai, T. Kanda: Development of an Interactive Humanoid Robot Robovie – An interdisciplinary approach, Robot. Res. **6**, 179–191 (2003)
- 59.98 K. Nickel, R. Stiefelwagen: Pointing gesture recognition based on 3 D-tracking of face, hands and head orientation, Proc. international conference on Multimodal interfaces (ICMI) (2003) pp. 140–146
- 59.99 M. Ito, J. Tani: Joint attention between a humanoid robot and users in imitation game, Proc. International Conference on Development and Learning (ICDL) (2004)
- 59.100 V.V. Hafner, F. Kaplan: Learning to interpret pointing gestures: experiments with four-legged autonomous robots. In: *Biomimetic Neural Learning for Intelligent Robots. Intelligent Systems, Cognitive Robotics, and Neuroscience*, ed. by S. Wermter, G. Palm, M. Elshaw (Springer, Berlin, Heidelberg 2005) pp. 225–234
- 59.101 C. Breazeal, D. Buchsbaum, J. Gray, D. Gatenby, B. Blumberg: Learning from and about Others: Towards Using Imitation to Bootstrap the Social Understanding of Others by Robots, Artificial Life **11**(1–2), 31–62 (2005)
- 59.102 P.F. Dominey, M. Alvarez, B. Gao, M. Jeambrun, A. Cheylus, A. Weitzenfeld, A. Martinez, A. Medrano: Robot command, interrogation and teaching via social interaction, Proc. IEEE-RAS International Conference on Humanoid Robots (Humanoids) (2005) pp. 475–480
- 59.103 A.L. Thomaz, M. Berlin, C. Breazeal: Robot Science Meets Social Science: An Embodied Computational Model of Social Referencing, Workshop Toward Social Mechanisms of Android Science (CogSci) (2005) pp. 7–17
- 59.104 C. Breazeal, L. Aryananda: Recognition of Affective Communicative Intent in Robot-Directed Speech, Autonomous Robots **12**(1), 83–104 (2002)
- 59.105 H. Bekkering, A. Wohlschlaeger, M. Gattis: Imitation of gestures in children is goal-directed, Quart. J. Exp. Psychol. **53A**(1), 153–164 (2000)
- 59.106 M. Nicolescu, M.J. Matarić: Task Learning Through Imitation and Human-Robot Interaction. In: *Imitation and Social Learning in Robots, Humans and Animals: Behavioural, Social and Communicative Dimensions*, ed. by K. Dautenhahn, C.L. Nehaniv (Cambridge Univ. Press, Cambridge 2007) pp. 407–424
- 59.107 J. Demiris, G. Hayes: Imitative Learning Mechanisms in Robots and Humans, Proc. European Workshop on Learning Robots, ed. by V. Klingspor (1996) pp. 9–16
- 59.108 P. Gaussier, S. Moga, J.P. Banquet, M. Quoy: From perception-action loop to imitation processes: a bottom-up approach of learning by imitation, Appl. Artif. Intell. **7**(1), 701–729 (1998)
- 59.109 M. Ogino, H. Toichi, Y. Yoshikawa, M. Asada: Interaction rule learning with a human partner based on an imitation faculty with a simple visuo-motor mapping, Robot. Auton. Syst. **54**(5), 414–418 (2006)
- 59.110 A. Billard, M. Matarić: Learning human arm movements by imitation: Evaluation of a biologically-inspired connectionist architecture, Robot. Auton. Syst. **37**(2), 145–160 (2001)
- 59.111 W. Erlhagen, A. Mukovski, E. Bicho, G. Panin, C. Kiss, A. Knoll, H. van Schie, H. Bekkering: Goal-directed imitation for robots: A bio-inspired approach to action understanding and skill learning, Robot. Auton. Syst. **54**(5), 353–360 (2006)
- 59.112 A. Chella, H. Dindo, I. Infantino: A cognitive framework for imitation learning, Robot. Auton. Syst. **54**(5), 403–408 (2006)
- 59.113 S. Calinon, A. Billard: Incremental Learning of Gestures by Imitation in a Humanoid Robot, Proc. ACM/IEEE International Conference on Human-Robot Interaction (HRI) (2007) pp. 255–262
- 59.114 F. Guenter, M. Hersch, S. Calinon, A. Billard: Reinforcement Learning for Imitating Constrained Reaching Movements, Adv. Robot. **21**(13), 1521–1544 (2007), Special Issue on Imitative Robots
- 59.115 R.H. Cuijpers, H.T. van Schie, M. Koppen, W. Erlhagen, H. Bekkering: Goals and means in action observation: A computational approach, Neural Networks **19**(3), 311–322 (2006)
- 59.116 M.W. Hoffman, D.B. Grimes, A.P. Shon, R.P.N. Rao: A probabilistic model of gaze imitation and shared attention, Neural Networks **19**(3), 299–310 (2006)
- 59.117 Y. Demiris, B. Khadhour: Hierarchical attentive multiple models for execution and recognition of actions, Robot. Auton. Syst. **54**(5), 361–369 (2006)
- 59.118 J. Peters, S. Vijayakumar, S. Schaal: Reinforcement Learning for Humanoid Robotics, Proc. IEEE International Conference on Humanoid Robots (Humanoids) (2003)
- 59.119 T. Yoshikai, N. Otake, I. Mizuuchi, M. Inaba, H. Inoue: Development of an Imitation Behavior in Humanoid Kenta with Reinforcement Learning Algorithm based on the Attention during Imitation, Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2004) pp. 1192–1197

- 59.120 D.C. Bentivegna, C.G. Atkeson, G. Cheng: Learning Tasks from Observation and Practice, *Robot. Auton. Syst.* **47**(2–3), 163–169 (2004)
- 59.121 Y.K. Hwang, K.J. Choi, D.S. Hong: Self-Learning Control of Cooperative Motion for a Humanoid Robot, *Proc. IEEE International Conference on Robotics and Automation (ICRA)* (2006) pp. 475–480
- 59.122 A. Billard, K. Dautenhahn: Grounding communication in autonomous robots: an experimental study, *Robot. Auton. Syst.* **24**(1–2), 71–79 (1998), Special Issue on Scientific methods in mobile robotics
- 59.123 S. Schaal: Is Imitation Learning the Route to Humanoid Robots?, *Trends Cognit. Sci.* **3**(6), 233–242 (1999)
- 59.124 J.H. Maunsell, D.C. Van Essen: Functional properties of neurons in middle temporal visual area of the macaque monkey. II. Binocular interactions and sensitivity to binocular disparity, *Neurophysiol.* **49**(5), 1148–1167 (1983)
- 59.125 M. Arbib, T. Iberall, D. Lyons: Coordinated Control Program for Movements of the Hand, *Exp. Brain Res. Suppl.* **10**, 111–129 (1985)
- 59.126 D. Sternad, S. Schaal: Segmentation of endpoint trajectories does not imply segmented control, *Exp. Brain Res.* **124**(1), 118–136 (1999)
- 59.127 R.S. Sutton, S. Singh, D. Precup, B. Ravindran: Improved switching among temporally abstract actions, *Advances in Neural Information Processing Systems (NIPS)* **11**, 1066–1072 (1999)
- 59.128 Y. Demiris, G. Hayes: Imitation as a Dual Route Process Featuring Predictive and Learning Components: a Biologically-Plausible Computational Model. In: *Imitation in Animals and Artifacts*, ed. by K. Dautenhahn, C. Nehaniv (MIT Press, Cambridge 2002) pp. 327–361
- 59.129 D.M. Wolpert, M. Kawato: Multiple paired forward and inverse models for motor control, *Neural Networks* **11**(7–8), 1317–1329 (1998)
- 59.130 C. Nehaniv, K. Dautenhahn: *Imitation in Animals and Artifacts* (MIT Press, Boston 2002)
- 59.131 R.S. Sutton, A.G. Barto: Reinforcement learning: an introduction. In: *Adaptive computation and machine learning*, ed. by S. Editor (MIT Press, Cambridge 1998)
- 59.132 G. Rizzolatti, L. Fogassi, V. Gallese: Neurophysiological mechanisms underlying the understanding and imitation of action, *Nature Rev. Neurosci.* **2**, 661–670 (2001)
- 59.133 M. Iacoboni, R.P. Woods, M. Brass, H. Bekkering, J.C. Mazziotta, G. Rizzolatti: Cortical Mechanisms of Human Imitation, *Science* **286**, 2526–2528 (1999)
- 59.134 E. Oztotop, M. Kawato, M.A. Arbib: Mirror neurons and imitation: A computationally guided review, *Neural Networks* **19**(3), 254–321 (2006)
- 59.135 E. Sauser, A. Billard: Biologically Inspired Multimodal Integration: Interferences in a Human-Robot Interaction Game, *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2006) pp. 5619–5624
- 59.136 A.H. Fagg, M.A. Arbib: Modeling Parietal-Premotor Interactions in Primate Control of Grasping, *Neural Networks* **11**(7), 1277–1303 (1998)
- 59.137 E. Oztotop, M.A. Arbib: Schema Design and Implementation of the Grasp-Related Mirror Neuron System, *Biol. Cybernet.* **87**(2), 116–140 (2002)
- 59.138 M. Arbib, A. Billard, M. Iacoboni, E. Oztotop: Mirror neurons, Imitation and (Synthetic) Brain Imaging, *Neural Networks* **13**(8–9), 953–973 (2000)
- 59.139 E. Oztotop, M. Lin, M. Kawato, G. Cheng: Dexterous Skills Transfer by Extending Human Body Schema to a Robotic Hand, *Proc. IEEE-RAS International Conference on Humanoid Robots (Humanoids)* (2006) pp. 82–87
- 59.140 E. Sauser, A. Billard: Parallel and Distributed Neural Models of the Ideomotor Principle: An Investigation of Imitative Cortical Pathways, *Neural Networks* **19**(3), 285–298 (2006)
- 59.141 E.L. Sauser, A.G. Billard: Dynamic Updating of Distributed Neural Representations using Forward Models, *Biol. Cybernet.* **95**(6), 567–588 (2006)
- 59.142 M. Brass, H. Bekkering, A. Wohlschlaeger, W. Prinz: Compatibility between observed and executed movements: Comparing symbolic, spatial and imitative cues, *Brain Cognit.* **44**(2), 124–143 (2001)

Biologically Inspired

60. Biologically Inspired Robots

Jean-Arcady Meyer, Agnès Guillot

After having stressed the difference between bio-inspired and biomimetic robots, this chapter successively describes bio-inspired morphologies, sensors, and actuators. Then, control architecture that, beyond mere reflexes, implement cognitive abilities like memory or planning, or adaptive processes like learning, evolution and development are described. Finally, the chapter also reports related works on energetic autonomy, collective robotics, and biohybrid robots.

60.1 General Background	1395
60.2 Bio-inspired Morphologies	1396
60.3 Bio-inspired Sensors	1398
60.3.1 Vision	1398
60.3.2 Audition	1399
60.3.3 Touch	1400
60.3.4 Smell	1402
60.3.5 Taste	1402
60.3.6 Internal Sensors	1402
60.4 Bio-inspired Actuators	1402
60.4.1 Locomotion	1402
60.4.2 Grasping	1407
60.4.3 Drilling	1408
60.5 Bio-inspired Control Architectures	1408
60.5.1 Behavior-Based Robotics	1408
60.5.2 Learning Robots	1409
60.5.3 Evolving Robots	1410
60.5.4 Developing Robots	1411
60.6 Energetic Autonomy	1412
60.7 Collective Robotics	1413
60.8 Biohybrid Robots	1415
60.9 Discussion	1417
60.10 Conclusion	1418
References	1418

60.1 General Background

Human inventors and engineers have always found in Nature's products an inexhaustible source of inspiration. About 2400 years ago, for instance, Archytas of Tarentum allegedly built a kind of flying machine, a wooden pigeon balanced by a weight suspended from a pulley, and set in motion by compressed air escaping from a valve. Likewise, circa 105 AD, the Chinese eunuch Ts'ai Lun is credited with inventing paper, after watching a wasp create its nest. More recently, Antoni Gaudi's design of the still-unfinished Sagrada Família cathedral in Barcelona displays countless borrowings from mineral and vegetal exuberance.

Although a similar tendency underlied all attempts at building automata or protorobots up to the middle of the last century [60.1], in the last decades roboticists borrowed much more from mathematics, mechanics,

electronics, and computer science than from biology. On the one hand, this approach undoubtedly solidified the technical foundations of the discipline and led to the production of highly successful products, especially in the field of industrial robotics. On the other hand, it served to better appreciate the gap that still separates a robot from an animal, at least when qualities of autonomy and adaptation are sought. As such qualities are required in a continually growing application field – from planetary exploration to domestic uses – a spectacular reversal of interest towards living creatures can be noticed in current-day robotics, up to the point that it has been said that natural inspiration is the *new wave* of robotics [60.2].

Undoubtedly, this new wave would not have been possible without the synergies generated by recent

advances in biology – where so-called *integrative approaches* now produce a huge amount of data and models directly exploitable by roboticists – and in technology – with the massive availability of low-cost and power-efficient computing systems, and with the development of new materials exhibiting new properties. This will be demonstrated in this chapter, which first reviews recent research efforts in bio-inspired morphologies, sensors, and actuators. Then, control architectures that, beyond mere reflexes, implement cognitive abilities – like memory or planning – or adaptive processes – like learning, evolution and development – will be described. Finally, the chapter will also report related works on energetic autonomy, collective robotics, and biohybrid robots.

It should be noted that this chapter will describe both bio-inspired and biomimetic realizations. In fact, these two terms characterize, respectively, the extremi-

ties of a continuum in which, on the one side, engineers seek to reproduce some natural result, but not necessarily the underlying means, while, on the other side, they seek to reproduce both the result and the means. Thus, bio-inspired robotics tends to adapt to traditional engineering approaches some principles that are abstracted from the observation of some living creature, whereas biomimetic robotics tends to replace classical engineering solutions by as detailed mechanisms or processes that it is possible to reproduce from the observation of this creature. In practice, any specific application usually lies somewhere between these two extremities. Be that as it may, because biomimetic realizations are always bio-inspired, whereas the reverse is not necessarily true, qualifying expressions like bio-inspired or biologically inspired will be preferentially used in this chapter.

60.2 Bio-inspired Morphologies

Although not comparable to that of real creatures, the diversity of bio-inspired morphologies that may be found in the realm of robotics is nevertheless quite impressive. Currently, a huge number of robots populates the terrestrial, as well as aquatic and aerial, environments and look like animals as diverse as dogs, kangaroos, sharks, dragonflies, or jellyfishes, not to mention humans (Fig. 60.1).

In nature, the morphology of an animal fits its ecology and behavior. In robotics applications, bio-inspired morphologies are seldom imposed by functional considerations. Rather, as close a resemblance as possible to a given animal is usually sought per se, as in animatronics applications for entertainment industry. However, several other applications are motivated by the functional objective of facilitating human–robot interactions, thus allowing, for instance, children or elderly people to adopt artificial pets and enjoy their company. Such interactions are facilitated in the case of so-called *anthropopathic or human-friendly* robots, such as Kismet at MIT [60.3] or WE-4RII at Waseda University [60.4], which are able to perceive and respond to human emotions, and do themselves express apparent emotions influencing their actions and behavior (Fig. 60.2a,b).

Likewise, the Uando robot of Osaka University [60.5] is controlled by air actuators providing 43 degrees of freedom. The android can make facial ex-

pressions, eye, head, and body movements, and gestures with its arms and hands. Touch sensors with sensitivity to variable pressures are mounted under its clothing and silicone skin, while floor sensors and omnidirectional vision sensors serve to recognize where people are in order to make eye contact while addressing them during conversation. Moreover, it can respond to the content and prosody of a human partner by varying what it says and the pitch of its voice (Fig. 60.2c). See Chap. 58 for more references on human-friendly robots.

Another active research area in which functional considerations play a major role is that of shape-shifting robots that can dynamically reconfigure their morphology according to internal or external circumstances. Biological inspiration stems from organisms that can regrow lost appendages, like the tail in lizards, or from transitions in developmental stages, like morphogenetic changes in batrachians. For instance, the base topology of the Conro self-reconfigurable robot developed in the Polymorphic Robotics Laboratory at USC-ISI is simply connected as in a snake, but the system can reconfigure itself in order to grow a set of legs or other specialized appendages (Fig. 60.3) thanks to a dedicated hormone-like adaptive communication protocol [60.6, 7].

Chapter 39 is devoted to distributed and cellular robots and provides other examples of such reconfigurable robots.

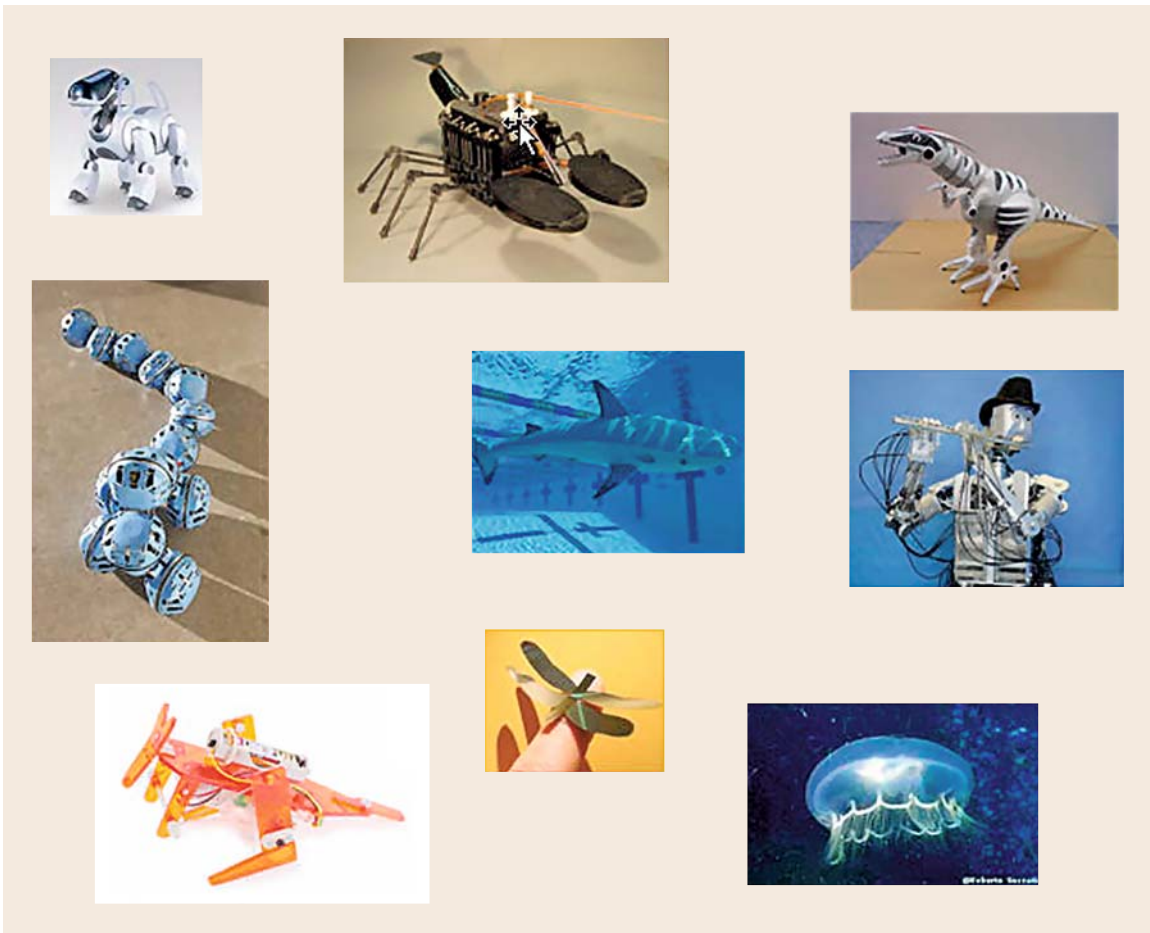


Fig. 60.1 A collection of zoomorphic robots

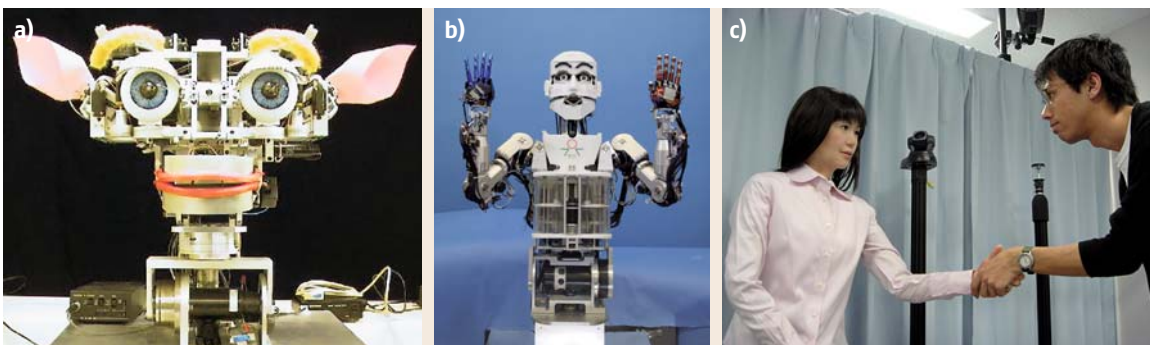


Fig. 60.2a–c Three humanoid robots. **(a)** Kismet ©Rodney Brooks, Computer Science and Artificial Intelligence Lab, MIT **(b)** WE-4RII ©Atsuo Takanishi Lab, Waseda University. **(c)** Uando ©Hiroshi Ishiguro, [ATR](#) Intelligent Robotics and Communication Lab, Osaka University

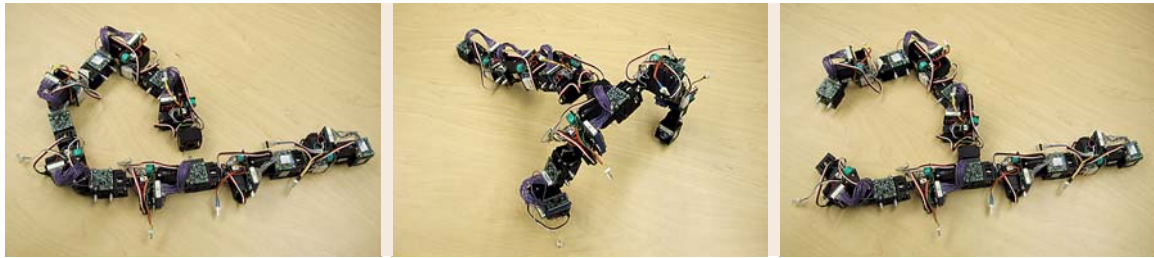


Fig. 60.3 A sequence reconfiguring a CONRO robot from a *snake* to a *T-shaped* creature with two legs. ©Wei-Min Shen, Polymorphic Robotics Laboratory, Univ. Southern California

60.3 Bio-inspired Sensors

60.3.1 Vision

Bio-inspired visual sensors in robotics range from very simple photosensitive devices, which mostly serve to implement phototaxis, to complex binocular devices used for more cognitive tasks like object recognition.

Phototaxis is seldom the focus of dedicated research. It is rather usually implemented merely to force a robot to move and exhibit other capacities such as obstacle avoidance or inter-robot communication.

Several visual systems calling upon optic-flow monitoring are particularly useful in the context of navigation tasks and are implemented in a variety of robots. This is the case with the work done in Marseilles' Biorobotics Laboratory that serves to understand how the organization of the compound eye of the housefly, and how the neural processing of visual information obtained during flight, endow this insect with various reflexes mandatory for its survival. The biological knowledge thus acquired

was exploited to implement optoelectronic devices allowing a terrestrial robot to wander in its environments while avoiding obstacles [60.8], or tethered aerial robots to track a contrasting target [60.9] or to automatically perform terrain-following, take-off, or landing [60.10] (Fig. 60.4).

The desert ant *Cataglyphis*, while probably merging optic-flow and odometry monitoring to evaluate its travel distances, is able to use its compound eyes to perceive the polarization pattern of the sky and infer its orientation. This affords it with accurate navigation capacities that make it possible to explore its desert habitat for hundreds of meters while foraging, and return back to its nest on an almost straight line, despite the absence of conspicuous landmarks and despite the impossibility of laying pheromones on the ground that would not almost immediately evaporate. Inspired by the insect's navigation system, mechanisms for path integration and visual piloting have been successfully

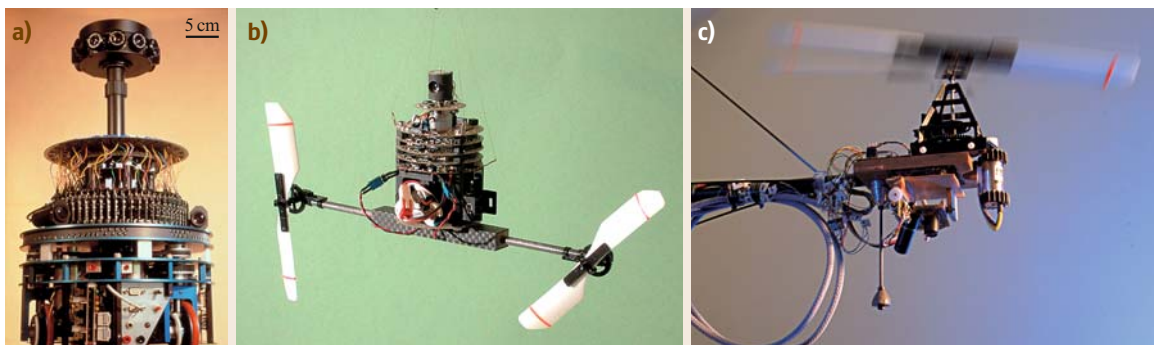


Fig. 60.4a–c Optoelectronic devices inspired by the housefly's compound eye. (a) Device for obstacle avoidance. (b) Device for target tracking. (c) Device for terrain following, take-off, and landing. ©CNRS Photothèque, Nicolas Franceschini, UMR6152 - Mouvement et Perception - Marseille

employed on mobile robot navigation in the Sahara desert [60.11].

Among the robotic realizations that are targeted at humanoid vision, some aim at integrating information provided by foveal and peripheral cameras. Ude et al. [60.12], in particular, describe a system that uses shape and color to detect and pursue objects through peripheral vision and then recognizes the object through a more detailed analysis of higher-resolution foveal images. The classification is inferred from a video stream rather than from a single image and, when a desired object is recognized, the robot reaches for it and ignores other objects (Fig. 60.5). Common alternatives to the use of two cameras per eye consist of using space-variant vision and, in particular, log-polar images. As an example, Metta [60.13] describes an attentional system that should be extended with modules for object recognition, trajectory tracking, and naive physics understanding during the natural interaction of a robot with the environment.

Other examples of robotic applications of perceptual processes underlying human vision are provided in Chap. 63 on perceptual robotics.

Vision-based simultaneous localization and mapping (SLAM) systems have also been implemented on humanoid robots, with the aim of increasing the autonomy of these machines. In particular, Davison et al. [60.14] used the HRP2 robot (Fig. 60.5) to demonstrate real-time SLAM capacities during agile combinations of walking and turning motions, using the robot's internal inertial sensors to monitor a type of three-dimensional odometry that reduced the local rate of increase in uncertainty within the SLAM map. The authors speculate that the availability of traditional odometry on all of the robot's degrees of freedom will allow more long-term motion constraints to be imposed and exploited by the SLAM algorithm, based on knowledge of possible robot configurations. Additional references to SLAM techniques are to be found in Chap. 37.

As another step towards autonomy in humanoid robots, mapping and planning capacities may be combined. Michel et al. [60.15], for instance, demonstrate that a real-time vision-based sensing system and an adaptive footstep planner allow a Honda ASIMO robot to autonomously traverse dynamic environments containing unpredictably moving obstacles.

60.3.2 Audition

Like vision, the sense of hearing in animals has been implemented on several robots to exhibit mere phono-

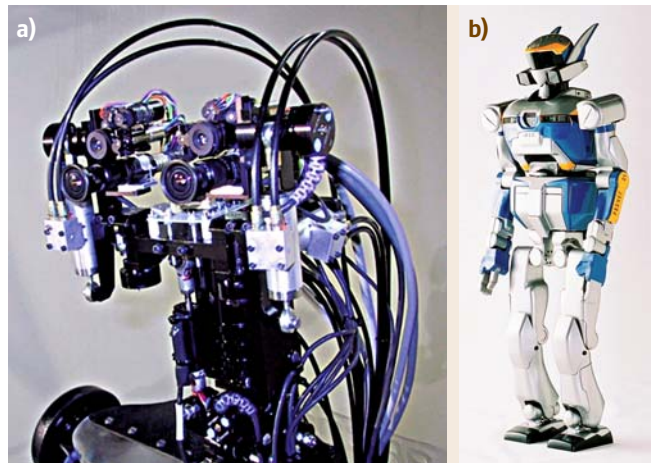


Fig. 60.5 (a) Four cameras implement foveal and peripheral vision in the head of the humanoid robot DB. Foveal cameras are above peripheral cameras. ©JST, ATR Robot developed by SARCOS. (b) The HRP2 humanoid robot ©Kawada Industries Inc./ National Institute of Advanced Industrial Science and Technology (AIST)

taxis behavior or more complex capacities such as object recognition.

At the University of Edinburgh, numerous research efforts are devoted to understanding the sensory-motor pathways and mechanisms that underlie positive or negative phonotaxis behavior in crickets through the implementation of various models on diverse robots such as the Khepera shown on Fig. 60.6. In particular, an analogue very-large-scale integrated (VLSI) circuit models the auditory mechanism that enables a female cricket to meet a conspecific male or to evade a bat (by the calling song or the echolocation calls they produce, respectively). The results suggest that the mechanism outputs a directional signal to sounds ahead at calling song frequency and to sound behind at echolocation frequencies, and that this combination of responses simplifies later neural processing in the cricket [60.16]. This processing is the subject of complementary modeling efforts in which spiking neuron controllers are also tested on robots, thus allowing the exploration of the functionality of the identified neurons in the insect, including the possible roles of multiple sensory fibers, mutually inhibitory connections, and brain neurons with pattern-filtering properties. Such robotic implementations also make the investigation of multimodal influences on the behavior possible, via the inclusion of an optomotor stabilization response and the demonstration that this may improve auditory tracking, particularly under conditions of random disturbance [60.17].

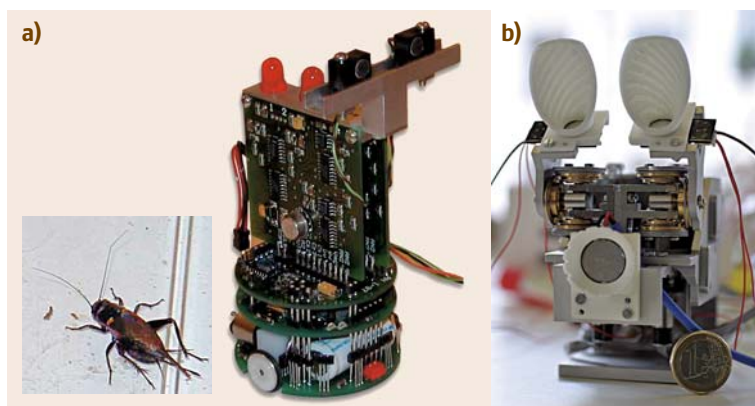


Fig. 60.6 (a) A Khepera robot equipped with a cricket-like auditory system. ©Barbara Webb, Institute for Perception, Action and Behaviour, University of Edinburgh. (b) The CIRCE robotic bat head ©Herbert Peremans, Active Perception Lab, Universiteit Antwerpen

Concerning more cognitive capacities, within the framework of the European Community (EC) project chiroptera-inspired robotic cephaloid (CIRCE), a bat head (Fig. 60.6) is used to investigate how the world is not just perceived, but actively explored, by bats. In particular, the work aims at identifying how various shapes, sizes, and movements influence the signals that the animal receives from its environment [60.18]. It is hoped that the principles gleaned from such work will prove useful in developing better antennas, particularly for wireless devices that are in motion and need to pick up complex signals from different directions.

Likewise, the Yale Sonar Robot, which is modeled after bat and dolphin echolocation behavior, is said to be so sensitive that it can tell whether a tossed coin has come up heads or tails. Called Rodolph – short for robotic dolphin – the robot is equipped with electrostatic transducers that can act either as transmitters or receivers to serve as the robot's mouth and ears. The design is inspired by bats, whose ears react by rotating in the direction of an echo source, and by dolphins, which appear to move around in order to place an object at a standard distance, thus reducing the complexity of object recognition [60.19]. Additional references to bio-inspired sonars are to be found in Chap. 21, dedicated to sonar sensing.

Nakadai et al. [60.20] describe a system that allows a humanoid robot to listen to a specific sound in under noisy environments (the human capability known as the *cocktail-party effect*) and to listen to several sources of speeches simultaneously, thus allowing it to cope with situations where someone or something playing sounds interrupts conversation (known as *barge-in* in spoken dialog systems). This system calls upon active motions directed at the sound source to improve localization by exploiting an *auditory fovea*. It also capitalizes on audio-

visual integration, thus making localization, separation, and recognition of three simultaneous speech sources possible.

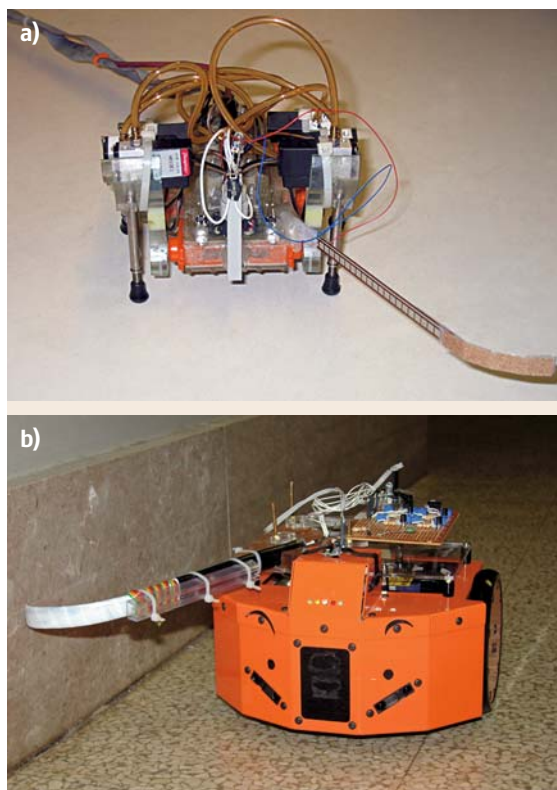


Fig. 60.7 (a) A simple antenna mounted on a Sprawlette robot ©Mark R. Cutkosky, Center for Design Research, Stanford University. (b) A more advanced tactile device ©Noah J. Cowan, Department of Mechanical Engineering, Johns Hopkins University

60.3.3 Touch

It is often asserted that, of all the five senses, touch is the most difficult to replicate in mechanical form. Be that as it may, a passive, highly compliant tactile sensor has been designed for the hexapedal running robot *Sprawlette* at Stanford, drawing inspiration from how the cockroach *Periplaneta americana* uses antenna feedback to control its orientation during a rapid wall-following behavior. Results on the stabilization of the robot suggest that the cockroach uses, at least in part, the rate of convergence to the wall – or *tactile flow* – to control its body orientation [60.21]. To make it possible to detect the point of greatest strain, or to differentiate between different shapes the sensor is bent into, more advanced versions of the antenna are currently under development (Fig. 60.7).

While a cockroach's antenna consists of multiple rigid segments and is covered along its length with

sensory receptors, a rat's whisker consists of a single, flexible, tapered hair and has tactile sensors located only at its base. The way in which two arrays of such sensors afford capacities of obstacle avoidance, texture discrimination, and object recognition has inspired several robotic realizations, notably that described by *Russel* and *Wijaya* [60.22] in which the whiskers are passive and rely upon the motion of the robot in order to scan the surface profile of touched objects. The robot is able to recognize a few objects formed from plane, cylindrical, and spherical surfaces. By using its simple manipulator, it can pick up and retrieve small objects.

Conversely, *Pearson et al.* [60.23] describe a touch system based on computational models of whisker-related neural circuitry in the rat brain, in which the whiskers will be actively scanning the surroundings. This work will contribute to the EC ICEA (integrating cognition, emotion, and autonomy – <http://www2.his.se/icea/>) project whose primary aim is to develop a cognitive systems architecture based on the anatomy and physiology of the mammalian brain.

In the field of humanoid robotics, investigations on touch sensors are being conducted at the University of Tokyo, where a robotic hand calling upon organic transistors as pressure sensors (Fig. 60.8a) has been produced. The same technology served to make a flexible artificial skin that can sense both pressure and temperature (Fig. 60.8b), thus more closely imitating the human sense of touch [60.24].

Another step in this direction has been made at the University of Nebraska [60.25], where a thin-film tactile sensor, which is as sensitive as the human finger in some ways, has been designed. When pressed against a textured object, the film creates a topographical map of the surface, by sending out both an electrical signal

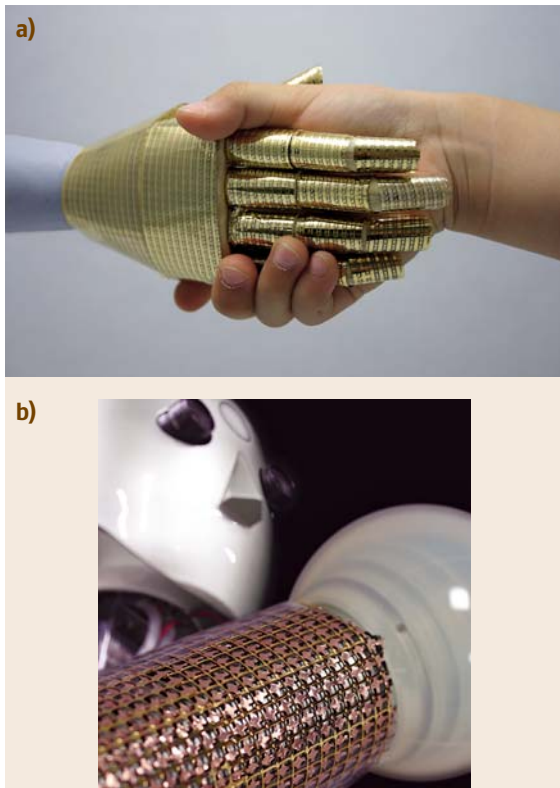


Fig. 60.8a,b Artificial skin devices at Tokyo University. (a) Pressure detection. (b) Pressure and temperature detection ©Takao Someya, Quantum-Phase Electronics Center, The University of Tokyo

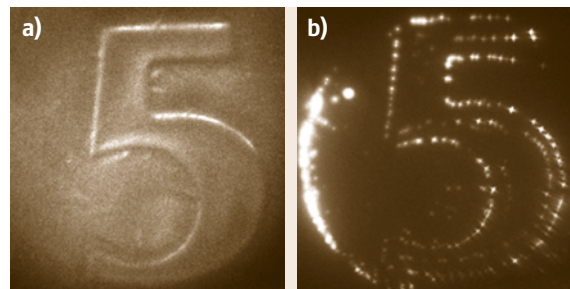


Fig. 60.9 (a) The optical image of a coin (b) The corresponding pressure image from the tactile sensor ©Ravi Saraf, Department of Chemical Engineering, University of Nebraska

and a visual signal that can be read with a small camera. The spatial resolution of these *maps* is as good as that achieved by human touch, as demonstrated by the image obtained when placing a coin on this mechanical fingertip (Fig. 60.9).

Although such sensors deal with texture in a way that is not at all like a fingertip, it has a high enough resolution to *feel* single cells, and therefore could help surgeons find the perimeter of a tumor during surgical procedures. Cancer cells, in particular breast cancer cells, have levels of pressure that are different from normal cells, and should feel *harder* to the sensor.

60.3.4 Smell

The way the nematode *Caenorhabditis elegans* uses chemotaxis – probably the most widespread form of goal-seeking behavior – to find bacterial food sources by following their odors has been investigated at the University of Oregon. This worm has a small nervous system (302 neurons), whose neurons and connectivity pattern have been completely characterized, so the neural circuit controlling chemotaxis is well known and, when implemented in a robot, proves to be able to cope with environmental variability and noise in sensory inputs [60.26]. The long-term objective of such work is to design a cheap, artificial eel that could locate explosive mines at sea. Among the research efforts that tackle the related and highly challenging issue of reproducing the odor-plume-tracking behavior in marine animals, results obtained on the RoboLobster are put in perspective in [60.27].

Other bio-inspired systems for odor recognition are under development in several places. For instance, the

chest of the humanoid WE-4RII robot of Waseda University (Fig. 60.2) is equipped with two mechanical lungs, each consisting of a cylinder and a piston, thanks to which the robot breathes air. Being also equipped with four semiconductor gas sensors, it recognizes the smells of alcohol, ammonia, and cigarette smoke [60.28].

60.3.5 Taste

A first robot with a sense of taste has recently been developed by NEC System Technologies, Ltd. Using infrared spectroscopic technology, this robot is capable of examining the taste of food and giving its name as well as its ingredients. Furthermore, it can give advice on the food and health issues based on the information gathered. The latest developments afford the robot with the capacity to distinguish good wine from bad wine, and Camembert from Gouda (<http://www.necst.co.jp/english/news/20061801/index.htm>).

60.3.6 Internal Sensors

Whereas the previous *external* sensors all provide information about an animal's or a robot's external world, *internal* sensors provide information about a creature's internal state. Although such so-called idiothetic sensors are widespread in robotic applications, measuring variables such as temperature, pressure, voltage, accelerations, etc., they are seldom biologically inspired, but in the implementation of a variety of visual-motor routines (smooth-pursuit tracking, saccades, binocular vergence, and vestibular-ocular and optokinetic reflexes), like those that are at work in the humanoid Cog robot mentioned later.

60.4 Bio-inspired Actuators

60.4.1 Locomotion

Crawling

Because they are able to move in environments inaccessible to humans, such as pipes or collapsed buildings, numerous snake-like robots have been developed for exploration and inspection tasks, as well as for participation in search-and-rescue missions. The Salamandra Robotica (Fig. 60.10) developed at the Ecole Polytechnique Fédérale de Lausanne (EPFL) in Switzerland extends these approaches because it is the first robot that combines the three modes of locomotion – serpentine

crawling, swimming, and walking – in a single robot. Being inspired by central pattern generators (CPG) found in vertebrate spinal cords, this work demonstrates how a primitive neural circuit for swimming, like the one found in the lamprey, can be extended by phylogenetically more recent limb oscillatory centers to explain the ability of salamanders to switch between swimming and walking. It also suggests a mechanism that explains how gait transition between swimming and walking can be obtained by simply varying the level of stimulation of the brainstem, and provides a potential explanation of how salamanders control their speed and direction

of locomotion, by modulating the level and the asymmetry, respectively, of the drive applied to the spinal cord [60.29].

Other applications are sought within the framework of the EC biomimetic structures for locomotion in the human body (BIOLOCH) project. In the perspective of helping doctors diagnose disease by carrying tiny cameras through patients' bodies, a robot designed to crawl through the human gut by mimicking the wriggling motion of an undersea worm has been developed by the project partners [60.30]. Drawing inspiration from the way polychaetes, or *paddle worms*, use tiny paddles on their body segments to push through sand, mud or water, they tackled the issue of supplying traditional forms of robotic locomotion that would not work in the peculiar environment of the gut (Fig. 60.10). The device is expected to lessen the chance of damaging a patient's internal organs with a colonic endoscope, and to enhance the exploration capacities afforded by *camera pills*.

Walking

Eight Legs. Joseph Ayers has developed a biomimetic robot based on the American lobster at the Marine Science Center of North Eastern University (Fig. 60.11). Capitalizing on recent advances in microcontrollers, smart materials, and microelectronic devices, this eight-legged ambulatory robot is intended for autonomous mine countermeasure operations in rivers, harbors, and/or the littoral zone of the ocean floor. Its control architecture supports a library of action patterns and reflexes – reverse-engineered from movies of lobsters behaving under the target conditions – that mediates tactile navigation, obstacle negotiation, and adaptation to surge.

The robot will have the overlying motivation to navigate on a specified compass heading. When encountering an obstacle, it will attempt to ascertain whether it is a mine candidate or not through dedicated sensors like an electronic nose, an acoustic hardness tester, or an active electric field perturbator. If the robot determines that the obstacle is not a mine candidate, it will decide whether to climb over the obstacle or to go around it using information supplied by its antennal sensors and claw-like surfaces. If climbing appears to be unfeasible, the robot will use a wall-following algorithm to go around the obstacle until it can resume its predetermined heading. This basic scenario will apply to almost all seafloor types because tactile queues from leg sensors will be used to determine whether the bottom is cobble, sand or hard, and because attitude reflexes will help with pitch and roll control [60.31].

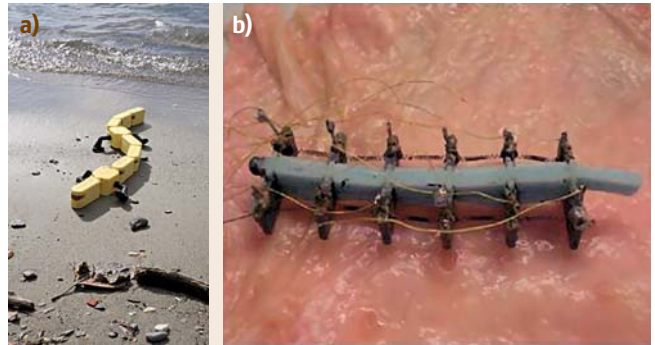


Fig. 60.10 (a) Salamandra Robotica, a robot that crawls, swims, and walks ©Photograph by A.Herzog, courtesy Biologically Inspired Robotics Group, EPFL (b) A worm-inspired robot designed to crawl through intestines ©Paolo Dario, Scuola Superiore Santa Anna, Pisa

Six Legs. In the performance energetics and dynamics of animal locomotion (PolyPEDAL) Laboratory at Berkeley, the observation that many animals self-stabilize to perturbations without a brain or its equivalent because control algorithms are embedded in their physical structure is widely exploited. Shape deposition manufacturing has allowed engineers to tune the legs of the SPRAWL family of hand-sized hexapedal robots inspired by the cockroach that are very fast (up to five body lengths per second), robust (hip-height obstacles), and that self-stabilize to perturbations without any active sensing [60.32]. One such robot is shown on Fig. 60.7. Capitalizing on previous work [60.33], a cricket-inspired robot, approximately 8 cm long, designed for both walking and jumping is under development at Case Western Reserve University, and is shown in Fig. 60.11. McKibben artificial muscles will actuate the legs, compressed air will be generated by an onboard power plant, and a continuous-time recurrent neural network will be used for control. Additionally, front legs will enable climbing over larger obstacles and will also be used to control the pitch of the body before a jump and, therefore, aim the jump for distance or height.

Water strider insects are able to walk on water because, instead of using buoyancy-like macroscale bodies, these very light and small creatures balance their weight using repulsive surface tension forces produced by the hydrophobic microhairs that cover their legs. In the NanoRobotics Laboratory of Carnegie Mellon University, Water Strider, a miniature microrobot, walks on water with legs made from hydrophobic Teflon®-coated wires and a body made of carbon fiber for minimal weight (Fig. 60.11). This tethered robot can successfully

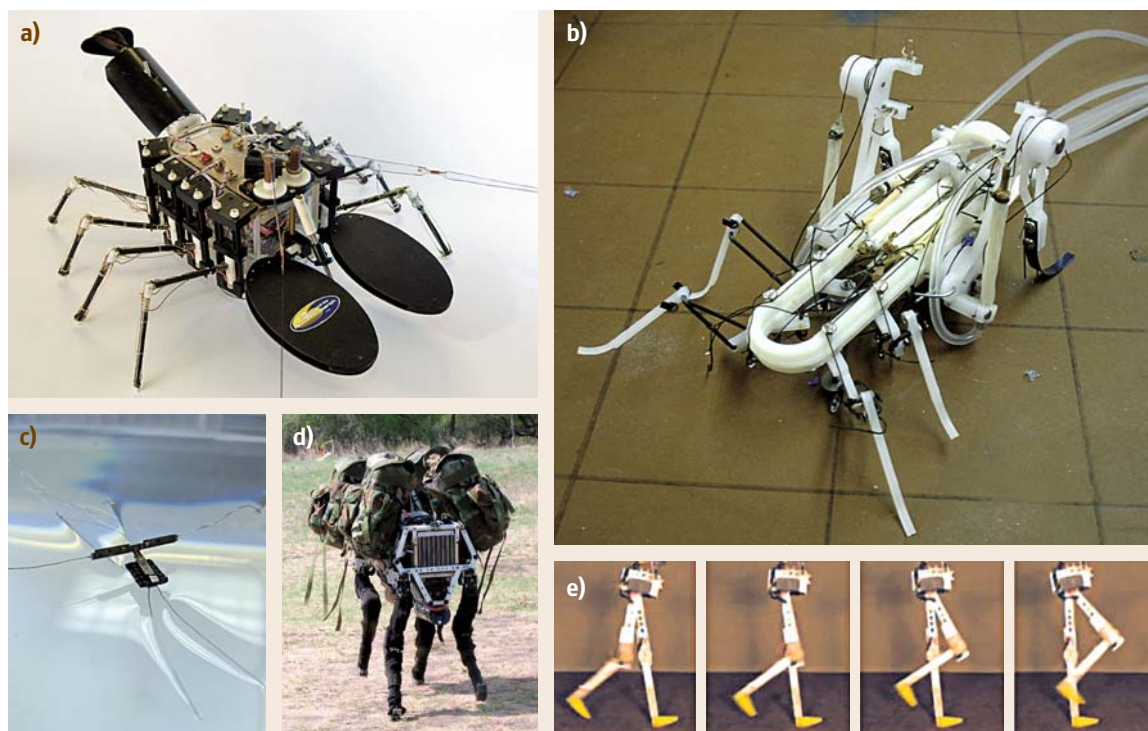


Fig. 60.11 (a) The lobster robot of Northeastern University ©Joseph Ayers, Department of Biology and Marine Science Center, Northeastern University. (b) The cricket robot from Case Western Reserve University ©Roger D. Quinn, Mechanical and Aerospace Engineering, Case Western Reserve University. (c) The Water Strider robot. ©Metin Sitti, NanoRobotics Lab, Carnegie Mellon University. (d) BigDog from Boston Dynamics ©Boston Dynamics, 2007. (e) RunBot from Stirling University ©Tao Geng, Department of Psychology University of Stirling.

move forward and backward, and can also make turns. Its maximum speed in forward motion is 2.3 cm/s. In the future, environmental monitoring applications on dams, lakes, sea, etc. would become possible using a network of these robots with miniature sensors and an onboard power source and electronics [60.34].

Four Legs. Engineers from Boston Dynamics claim to have developed *the most advanced quadruped robot on Earth* for the US Army. Called BigDog, it walks, runs, and climbs on rough terrain, and carries heavy loads. Being the size of a large dog or a small mule, measuring 1 m long and 0.7 m tall, and weighing 75 kg, BigDog has trotted at 5 km/h, climbed a 35° slope, and carried a 50 kg load so far. BigDog is powered by a gasoline engine that drives a hydraulic actuation system. Its legs are articulated like an animal's, and have compliant elements that absorb shock and recycle energy from one step to the next (Fig. 60.11). Another quadruped with amazing locomotion capabilities is Scout II, presumably

the world's first galloping robot, developed at McGill University [60.35]. Using a single actuator per leg – the hip joint providing leg rotation in the sagittal plane – and each leg having two degrees of freedom (DOF) – the actuated revolute hip DOF, and the passive linear compliant leg DOF – the system exhibits passively generated bounding cycles and can stabilize itself without the need of any control action. This feature makes simple open-loop control of complex running behaviors such as bounding and galloping possible.

Two Legs. Developed at Stirling University, RunBot is probably the world's fastest biped robot for its size. Being 30 cm high, it can walk at a speed of 3.5 leg-lengths per second, which is comparable to the fastest relative speed of human walking (Fig. 60.11). This robot has some special mechanical features, e.g., small curved feet allowing rolling action and a properly positioned center of mass, which facilitate fast walking through exploitation of its natural dynamics. It also calls upon

a sensor-driven controller that is built with biologically inspired sensor and motor-neuron models, the parameters of which can be tuned by a policy gradient reinforcement learning algorithm in real time during walking. The robot does not employ any kind of position or trajectory-tracking control algorithm. Instead, it exploits its own natural dynamics during critical stages of its walking gait cycle [60.36].

Additional references to legged robots can be found in Chap. 16.

Wall-Climbing

In the Biomimetic Dextrous Manipulation Laboratory at Stanford University, researchers are working on a gecko-like robot, called Stickybot, designed to climb smooth surfaces like glass without using suction or adhesives (Fig. 60.12). Geckos can climb up walls and across ceilings thanks to roughly half a million tiny hairs, or setae, on the surface of each of their feet and to the hundreds to thousands of tiny pads, or spatulae, at the tip of each hair. Each of these pads is attracted to the wall by intermolecular van der Waals forces, which allow the gecko's feet to adhere. Conversely, if the hair is levered upward at a 30° angle, the spatulae at the end of the hair easily detach. The gecko does this simply by peeling its toes off the surface. Inspired by such structures and mechanisms, the Stickybot's feet are covered with thousands of synthetic setae made of an elastomer. These tiny polymer pads ensure a large area of contact between the feet and the wall, thus maximizing the expression of intermolecular forces. In the same laboratory, a six-legged robot called Spinybot climbs vertical surfaces according to similar principles. Spinybot's feet and toes are made from several different polymers, which range from flexible to rigid, thus enabling the robot to absorb jolts and bumps, much as animals' feet do [60.37].

The robots in scansorial environments (RiSE) project funded by the Defense Advanced Research Projects Agency (DARPA) Biodynamics program constitutes an extension of these research efforts that aims at building a bio-inspired climbing robot with the unique ability to walk on land and climb on trees, fences, walls, as well as other vertical surfaces. It calls upon novel robot kinematics, precision-manufactured compliant feet and appendages, and advanced robot behaviors [60.38, 39].

Jumping

In the perspective of environment exploration and monitoring, Scarfogliero et al. [60.40] describe a lightweight microrobot that demonstrates that jumping can be more

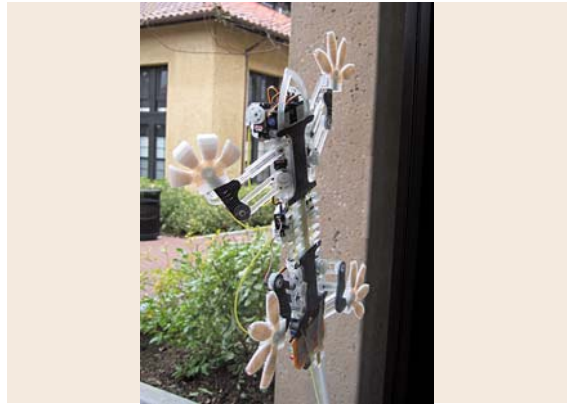


Fig. 60.12 Stickybot, the artificial gecko ©Mark R. Cutkosky, Stanford University Center for Design Research

energetically efficient than just walking or climbing, and can be used to overcome obstacles and uneven terrains. During the flight phase, energy from an electric micro-motor is collected in the robot's springs, and is released by a click mechanism during take-off. In this way the instantaneous power delivered by the rear legs is much higher than that provided by the motor.

Swimming

Several biomimetic robots are being produced that emulate the propulsive systems of fish, dolphins, or seals, and exploit the complex fluid mechanics these animals use to propel themselves. A primary goal of these projects is to build machines that can maneuver by taking advantage of flows and body positions, leading to huge energy savings, and substantially increasing the length of swimming time. For instance, the group at MIT Towing Tank designed two robotic fish, *RoboTuna*

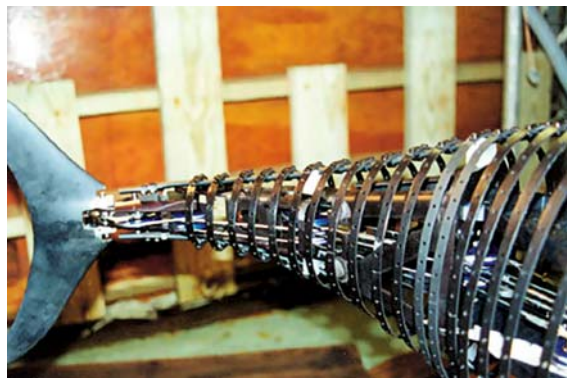


Fig. 60.13 The tail of RoboTuna at MIT ©Michael Triantafyllou, MIT Towing Tank

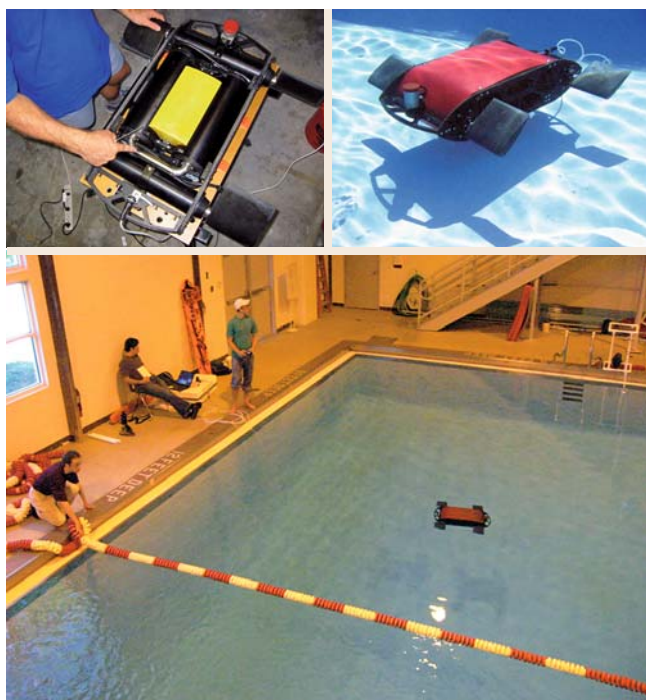


Fig. 60.14 The robot Madeleine ©John H. Long, Jr, Interdisciplinary Robotics Research Laboratory, Vassar College

and *RoboPike*, which use servo motors and spring element spines (Fig. 60.13), and serve to demonstrate the advantages of flapping foil propulsion. It has thus been shown that RoboTuna can reduce its drag by more than 70% compared to the same body towed straight and

rigid [60.41]. Likewise, it appears that biomimetic fish can turn at a maximum rate of $75^\circ/\text{s}$, whereas conventional rigid-bodied robots and submarines turn at approximately $3\text{--}5^\circ/\text{s}$ [60.42].

The robot Madeleine of Vassar College imitates the design of a turtle. Measuring 80×30 cm and weighing 24 kg, it has a comparable power output, and its polyurethane flippers have the same stiffness as a real turtle's, but the latter are operated by electric motors connected to an onboard computer (Fig. 60.14). Because it can swim underwater using four flippers, like many extinct animals, or with two flippers, like modern animals, this robot has been used to investigate theories of locomotion in existing and extinct animals. It thus appears that having four flippers does not improve the top speed – apparently because the front flippers create turbulence that interferes with the rear flippers' ability to generate forward propulsion – but does increase energy use. This may explain why natural selection favored two-flipper over four-flipper creatures like the plesiosaurs, and why four-flipper animals such as penguins, sea turtles, and seals use only two of their limbs for propulsion [60.43].

Flying

Flapping wings offer several advantages over the fixed wings of today's reconnaissance drones, like flying at low speeds, hovering, making sharp turns, and even flying backward. Like in animals, the vortex created beneath each wing is exploited to create the push necessary for robots to take to the sky.

The goal of the Micromechanical Flying Insect (MFI) project at Berkeley is to develop a 25 mm robot capable of sustained autonomous flight, which could be used in search, rescue, monitoring, and reconnaissance. Such tiny robots will be based on biomimetic principles that capture some of the exceptional flight performance achieved by true flies, i.e., large forces generated by non-steady-state aerodynamics, a high power-to-weight ratio motor system, and a high-speed control system with tightly integrated visual and inertial sensors. Design analysis suggests that piezoelectric actuators and flexible thorax structures can provide the necessary power density and wing stroke, and that adequate power can be supplied by lithium batteries charged by solar cells. Likewise, mathematical models capitalizing on wing-thorax dynamics, flapping flight aerodynamics in a low-Reynolds-number regime, body dynamics, as well as on a biomimetic sensory system consisting of ocelli, halteres, magnetic compass, and optical flow sensors, have been used to generate realistic simulations for MFI

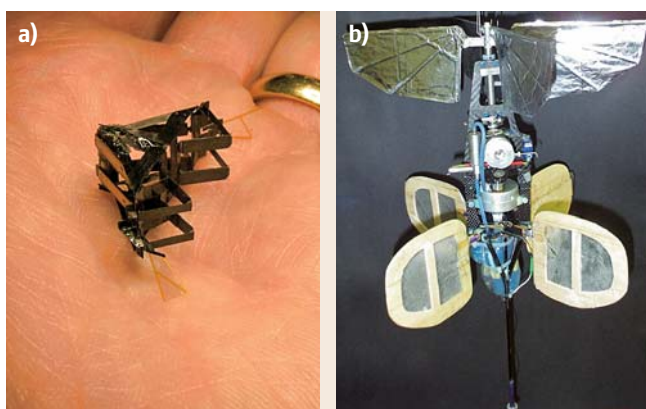


Fig. 60.15a,b Flapping-wing robots. (a) MFI ©Ronald Fearing, Berkeley's Electrical Engineering and Computer Science Department. (b) Mentor ©James DeLaurier, Institute for Aerospace Studies, University of Toronto

and insect flight. In turn, such simulations served to design a flight control algorithm maintaining stable flight in hovering mode [60.44,45]. A first MFI platform, which flaps its two wings and is the right size, has already been produced (Fig. 60.15).

The four-winged ornithopter Mentor (Fig. 60.15), which is being developed at the University of Toronto as part of a general research effort targeted at flapping-wing flight [60.46,47], is said to be the first artificial flapping-wing device that successfully hovered, doing so with the agility of a hummingbird. In particular, it exhibited the *clap-fling* behavior that the animal uses to enhance lift by clapping its wings together, then flinging them apart at high speeds. This augments the lift-producing bound vorticity on the wings. Likewise, the aircraft demonstrated the capability to shift from hovering to horizontal flight, which inspires current research. Mentor is about 30 cm long and weighs about 0.5 kg, but engineers eventually hope to shrink it to hummingbird size and weight. Other comparable micro aerial vehicle (MAV) devices are reported in [60.48,49].

On a much larger scale, a few manned flapping-wing robots have also been designed. In Votkinsk in the

1990s, *Toporov* built a tow-launched biplane ornithopter that could reportedly be made to climb and fly for 200 m as a result of the pilot's muscular effort [60.50]. More recently, within the Ornithopter project of SRI International and the University of Toronto [60.51], the two-winged Flapper plane has flown for 14 s at an average speed of 88 km/h. It has a 12 m wingspan and weighs 350 kg with pilot and fuel. The wings are made of carbon fiber and Kevlar, and are moved by a gas-powered engine. A description of previous attempts at making such a platform fly is available in [60.52].



Fig. 60.16 The OCTARM robot ©Ian Walker, Department of Electrical and Computer Engineering, Clemson University

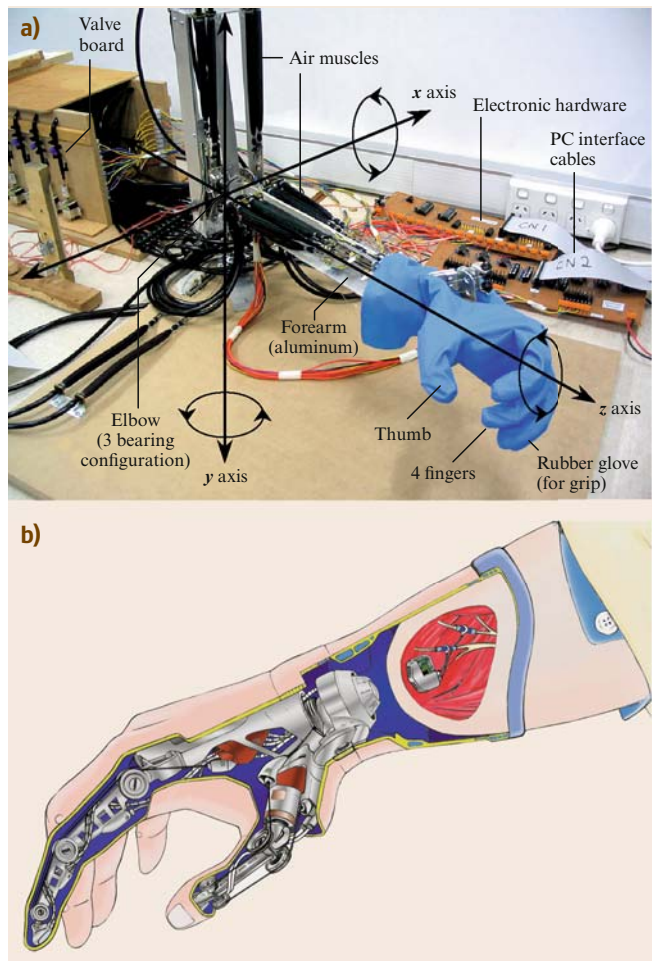


Fig. 60.17 (a) The humanoid hand of Curtin University of Technology. ©Euan Lindsay and Peter Scarfe, Mechatronic Engineering, Curtin University of Technology (b) The Cyberhand project ©Project Cyberhand (IST-2001-35094) and Scuola Superiore Santa Anna, Pisa

60.4.2 Grasping

When hunting and grabbing food, the octopus uses all the flexibility its arms are capable of. However, when feeding, the animal is able to bend its flexible arms to form *joints* like those in human arms. Inspired by such dexterous appendages found in cephalopods – particularly the arms and suckers of octopus, and the arms and tentacles of squid – Walker et al. [60.53] describe recent results in the development of a new class of soft, continuous-backbone robot manipulators. Fed by fundamental research into the manipulation tactics, sensory biology, and neural control of octopuses, this work in turn is leading to the development of artificial devices based on both electroactive polymers and pneumatic McKibben muscles, as well as to novel approaches to motion planning and operator interfaces for the so-called OCTARM robot (Fig. 60.16). Likewise, inspired by biological trunks and tentacles, a multisection continuum robot, Air-Octor, in which the extension of each section can be independently controlled, exhibits both bending and extension capacities, and demonstrates superior performance arising from the additional degrees of freedom than arms with comparable total degrees of freedom [60.54].

Human grasping has inspired the humanoid hand developed at Curtin University of Technology by Scarfe and Lindsay [60.55]. The corresponding system presents ten individually controllable degrees of freedom ranging from the elbow to the fingers, and is actuated through 20 McKibben air muscles, each supplied by a pneumatic pressure-balancing valve that allows for proportional control to be achieved with simple and inexpensive components. The hand is able to perform a number of

human-equivalent tasks, such as grasping and relocating objects (Fig. 60.17). A similar research is funded by the EC CYBERHAND project that aims at developing a cybernetic prosthetic hand [60.56]. It is hoped that the device will recreate the *lifelike* perception of the natural hand, and thus increase its acceptability. To this end, biomimetic sensors replicating the natural sensors are to be developed, and dedicated electrodes – capable of delivering sensory feedback to the amputee's central nervous system and extracting his intentions – are to be designed (Fig. 60.17).

Chapter 15 provides additional references to robot hands.

60.4.3 Drilling

Due to the ultraviolet flux in the surface layers of most bodies in the solar system, future astrobiological research is increasingly seeking to conduct subsurface penetration and drilling to detect the chemical signature for extant or extinct life. To address this issue, Gao et al. [60.57] present a bio-inspired micropenetrator implementing a novel concept of two-valve-reciprocating motion that is inspired by the way a wood wasp uses its ovipositor to drill holes into trees in order to lay its eggs. Indeed, such ovipositors can be split into two longitudinal halves, one side being equipped with cutting teeth and the other with pockets that serve to carry the sawdust away from the hole. The cutting teeth are used to cut the wood in compression and avoid buckling. The sawdust they produce is deposited into the pockets and carried to the surface on the upstroke. The two sides repeat this process in a reciprocating motion. The corresponding artificial system is lightweight (0.5 kg), driven at low power (3 W), and able to drill deep (1–2 m).

60.5 Bio-inspired Control Architectures

Attempts at tackling the *whole iguana* challenge [60.58], i. e., that of integrating sensors, actuators and control in the design of a simple but complete artificial animal, are abundant in the literature and several of the aforementioned realizations come under this objective. However, the corresponding controllers usually implement mere reflexes that serve to cope with present circumstances only. In this paragraph, more cognitive architectures, able to deal with past and future events as well, and in which adaptive mechanisms such as learning, evolution, and development may be incorporated, will be mentioned.

60.5.1 Behavior-Based Robotics

Under the aegis of so-called *behavior-based* robotics – to which Chap. 38 is dedicated – many systems with minimally cognitive architectures have been developed. For instance, the series of robots designed by Brooks and his students at MIT demonstrate that the *subsumption architecture* [60.59] may endow artificial animals with adaptive capacities that do not necessitate high-level reasoning [60.60]. Moreover, there are some indications that such control architecture may be at work in real animals, such as the coastal snail *Littorina* [60.61].

Likewise, the *schemas* that are used by Arkin and his students at the Georgia Institute of Technology to control numerous other robots [60.62] have roots in psychology [60.63] and neuroscience [60.64].

60.5.2 Learning Robots

Different bio-inspired learning mechanisms – like those implementing associative, reinforcement, or imitation learning schemes – are currently at work in robotic applications.

For instance, in the robotics laboratory at Nagoya University, the robot Brachiator is able to swing from handhold to handhold like a gibbon (Fig. 60.18). The robot is equipped with legs that generate initial momentum, and with a computer vision system to figure out where to place its hand-like grippers. A standard reinforcement learning algorithm is used to learn the right sensory–motor coordination required to move along a horizontal scale while hanging on successive rungs: it provides a punishment signal when the robot misses the next handhold, and a reward signal when it succeeds. Thus, after a number of failed trials, the robot eventually succeeds in safely moving from one extremity of the scale to the other [60.65].

Bio-inspired associative learning mechanisms are used in applications that capitalize upon the place cells and head-direction cells found in hippocampal and para-hippocampal structures in the brain to implement map-building, localization, and navigation capacities in robots (see [60.66–68] for reviews). Likewise, reinforcement learning mechanisms inspired by the presumed function of dopaminergic neurons [60.69] may be associated with models based on the anatomy and physiology of basal ganglia and related structures [60.70,71], which endow a robot with a motivational system and action-selection capacities – i. e., those of deciding when to shift from one activity to another, according to the various subgoals the unexpected events encountered during the fulfillment of a given mission generate. Such controllers and capacities are currently combined in the Psikharpax artificial rat that will be able to explore an unknown environment, to build a topological map of it, and to plan trajectories to places where it will fulfill various internal needs, like *eating*, *resting*, *exploring* or *avoiding danger* [60.72] (Fig. 60.18) – as a contribution to the EC project ICEA mentioned before.

MirrorBot, another EC project [60.73], capitalizes on the discovery of mirror neurons in the frontal lobes of monkeys, and on their potential relevance to human brain evolution. Indeed, mirror neuron areas correspond

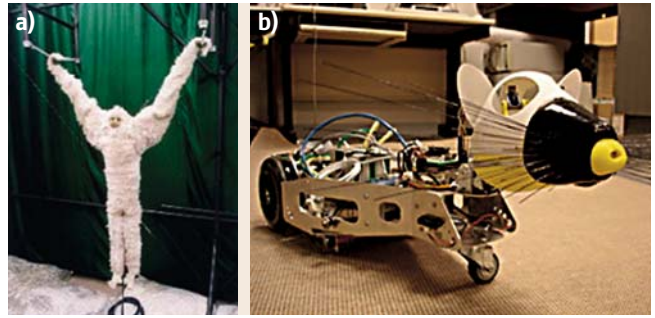


Fig. 60.18 (a) Brachiator, the artificial gibbon ©Toshio Fukuda, Department of Micro-Nano Systems Engineering, Nagoya University. (b) Psikharpax, the artificial rat ©Christophe Grand, Steve N’Guyen, Patrick Pirim, Bruno Gas, Ryad Benosman, ISIR, Université Paris 6

to cortical areas that are related to human language centers, and it seems that these neurons play a critical role in cortical networks establishing links between perception, action, and language [60.74]. The project has developed an approach of biomimetic multimodal learning, including imitation learning, using a mirror-neuron-based robot, and has investigated the task of foraging for objects that are designated by their names [60.75].

At the Neuroscience Institute in San Diego, a series of brain-based devices (BBDs) – i. e., physical devices with simulated nervous systems that guide behavior, to serve as a heuristic for understanding brain function – have been constructed. These BBDs are based on biological principles and alter their behavior to the environment through self-learning. The resulting systems autonomously generalize signals from the environment into perceptual categories and become increasingly successful in coping with the environment. Among these devices, the robot Darwin VII is equipped with a charge-coupled device (CCD) camera for vision, microphones for hearing, conductivity sensors for taste, and effectors to move its base and its head, and with a gripping manipulator having one degree of freedom. Its control architecture is made of 20 000 neurons, and it is endowed with a few instincts, like an interest in bright objects, a predilection for tasting things, and an innate notion of what tastes good. Thus, the robot explores its environment and quickly learns that striped blocks are yummy and that spotted ones taste bad. Based on the same robotic platform, Darwin VIII is equipped with a simulated nervous system containing 28 neural areas, 53 450 neuronal units, and approximately 1.7 million synaptic connections. It demonstrates that different brain areas and modalities can yield a coherent perceptual re-

sponse in the absence of any superordinate control, thus solving the so-called *binding* problem. In particular, the robot binds features such as colors and line segments into objects and discriminates between these objects in a visual scene [60.76]. Darwin IX is a mobile physical device equipped with artificial whiskers and a neural simulation based on the rat somatosensory system. Neuronal units with time-lagged response properties, together with the selective modulation of neural connection strengths, provide a plausible neural mechanism for the spatiotemporal transformations of sensory input necessary for both texture discrimination and selective conditioning to textures. Having an innate tendency to avoid *foot-shock* pads made of reflective construction paper deposited on the ground of its experimental arena, the

robot may be conditioned to avoid specific textures encountered near these aversive stimuli [60.77]. Darwin X incorporates a large-scale simulation of the hippocampus and surrounding areas, thus making it possible to solve a dry version of the Morris *water-maze* task, in which the robot must find a hidden platform in its environment using only visual landmarks and self-movement cues to navigate to the platform from any starting position [60.78]. Besides its ability to learn to run mazes like rats, Darwin X has been thrown into a soccer match, and turned out to be victorious in the 2005 RoboCup [US Open](#). Finally, Darwin XI combines the main characteristics of several previous versions, including a whisker system, and serves to demonstrate the robot's capacity to learn the reward structure of the environment, as well as the reversal of behavior when this structure changes.

In the perspective of exploring the role that chaotic dynamics may play in self-organizing behavior, researchers involved in the [NASA-funded](#) self-organizing dynamically adaptable systems (SODAS) project are using a nonlinear dynamics approach to model how the brain, which is usually in a high-dimensional, disorderly *basal* state, instantly shifts from a chaotic state to an attractor four or five times a second in order to recognize something familiar, or to make a decision. Such phase transitions and attractors in one area of the brain affect attractors in other areas, and are considered to produce intentional behavior. Focused on the way the brain orients the body in space and uses positive and negative reinforcement from the environment to navigate autonomously to a destination, the goal of the SODAS project is to enable robots to do the same on future National Aeronautics and Space Administration ([NASA](#)) missions. In particular, it has produced the KIV architecture that models the brain's limbic system, the simplest neurological structure capable of acting intentionally in an inherently self-consistent manner. [Kozma et al.](#) [60.79] describe how, in a two-dimensional computer simulation of a Martian landscape, KIV uses positive and negative reinforcement to learn the most effective path to a goal, and uses habituation to reduce the distraction of ambient noise and other irrelevant sensory inputs.

Other bio-inspired approaches to the design of control architectures are to be found in Chap. 62, which is dedicated to neurorobotics.

60.5.3 Evolving Robots

Using appropriate evolutionary algorithms and artificial selection processes to adapt from generation to gener-

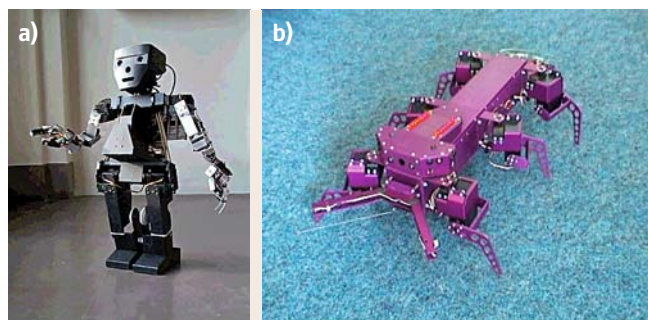


Fig. 60.19 (a) The robot Elvis. ©William Langdon, Mathematical Sciences, University of Essex (b) The robot SECT ©Jérôme Kodjabachian et David Filliat, AnimatLab et LIP6, Université Paris 6

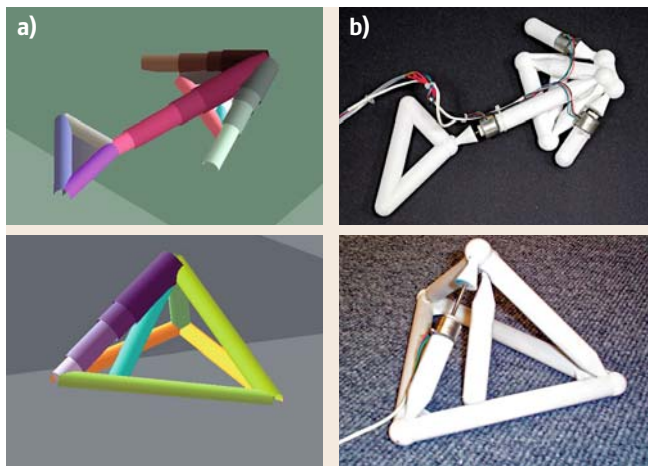


Fig. 60.20 (a) Simple robots whose morphology and control were evolved in simulation. (b) The corresponding physical realizations obtained through rapid-prototyping technology ©Jordan Pollack, Computer Science Department, Brandeis University

ation the code that describes a robot's controller has become current practice. Usually, an efficient code is sought in simulation and then implemented on a real robot [60.80].

At Chalmers University of Technology, for example, such an approach has been used to coordinate the visual information acquired through the two eyes of the humanoid robot Elvis (Fig. 60.19a) with the motor orders sent to its effectors. Thus the robot is able to track and point a visual target [60.81]. In a similar manner, at the Paris VI University, an incremental approach capitalizing upon solutions to simpler problems to devise solutions to more complex ones has been applied to the evolution of neural controllers for locomotion and obstacle avoidance in the six-legged SECT robot (Fig. 60.19b) [60.82]. At the same university, simulated artificial evolution has been applied to the control of horizontal flight in an artificial bird, or to slope-soaring in a glider [60.83]. The corresponding controllers will be implemented on real platforms as a contribution to the Robur project [60.84].

For the genetically organized lifelike electromechanics (Golem) project at Brandeis University, *Lipson and Pollack* [60.85] went beyond the evolution of hardware controllers and demonstrated for the first time a path that allows the transfer of virtual diversity of morphology into reality. They thus conducted a set of experiments in which simple electromechanical systems composed of thermoplastic, linear actuators, and neurons evolved from scratch to yield physical locomoting machines (Fig. 60.20).

Additional references to evolutionary robotics can be found in Chap. 61.

60.5.4 Developing Robots

Two varieties of developmental processes are currently applied to robotics. The first is related to evolution and aims at designing indirect coding schemes which, instead of directly specifying a robot's behavior and/or shape, describe developmental rules according to which complex neural controllers and/or morphologies can be derived from simple programs (see [60.86] for a review). This is done in the hope that the approach will scale up with the complexity of the control problems to be solved. Such a methodology has been applied at the Paris VI University to evolve neural controllers for diverse rolling, walking, swimming, and flying animats or robots [60.87].

The second series of developmental process (see [60.88] for a review) are related to learning and aim

at reproducing the successive sensory-motor and cognitive stages exhibited by developing animals, especially children [60.89]. As an example of such an endeavor, the upper-torso humanoid robot called Cog [60.90] (Fig. 60.21a), developed at MIT, has 22 DOFs and a variety of external and internal sensory systems, including visual, auditory, vestibular, kinesthetic, and tactile senses [60.91]. It has been endowed with various basic drives provided by its primary designers and, like a human baby, it has gone through a series of parallel developmental stages in its sensory-motor and cognitive capabilities. Among the results already acquired are the development of mechanisms for reaching and grasping, for rhythmic movement control, for

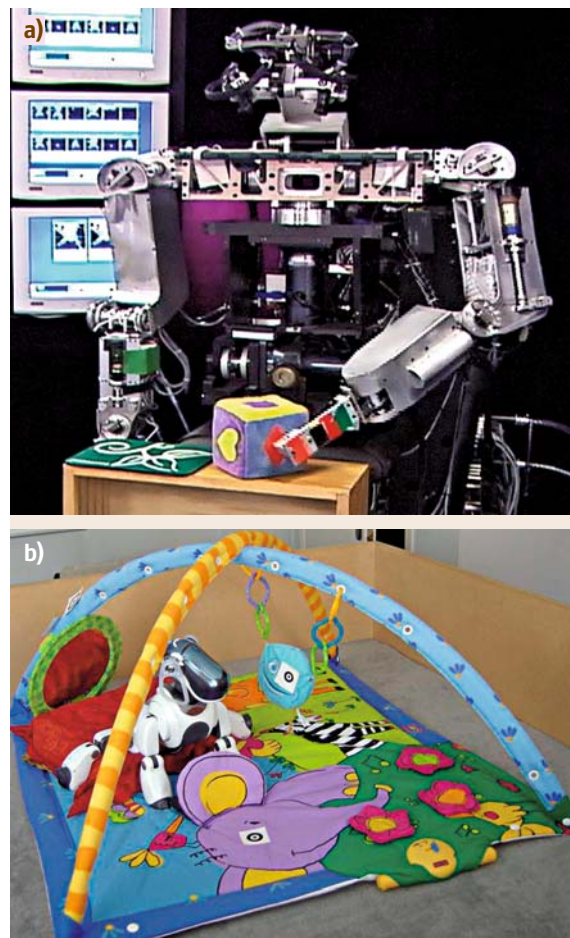


Fig. 60.21 (a) The Cog robot. ©Rodney Brooks, Computer Science and Artificial Intelligence Lab, MIT. (b) An Aibo robot in a playground environment ©Frédéric Kaplan, Sony Computer Science Laboratory, Paris

visual search and attention, for imitation learning, for emotional regulation of social dynamics, for saliency identification through shared attention, and for the emergence of a theory of mind [60.92]. The latter expression is commonly used to design the set of cognitive skills that make it possible to attribute beliefs, goals, and desires to other individuals. In a similar perspective, at Sony Computer Science Laboratory (CSL) in Paris, a mechanism of so-called *intelligent adaptive curiosity* serves as a source of self-development for an Aibo

robot placed in a playground environment (Fig. 60.21b) that tries to maximize its learning progress. According to this mechanism, the robot focuses on situations which are neither too predictable nor too unpredictable, and the complexity of its activity autonomously increases with time. In particular, it first spends time in situations which are easy to learn, then progressively shifts its attention to situations of increasing difficulty, avoiding situations in which nothing new can be learned [60.93].

60.6 Energetic Autonomy

The majority of bio-inspired systems described so far were targeted at increasing the robots' behavioral autonomy. However, a second, even more challenging, issue remains to be tackled, that of reproducing the energetic autonomy of animals, and the way they manage to discover and exploit resources to supply their energy needs. Very few attempts have been made in this direction yet. As a notable exception, Chew Chew was a 12-wheeled, train-like robot developed at the University of South Florida, which derived power through a microbial fuel cell (MFC) stomach (Fig. 60.22a). The stomach broke down refined sugar using *Escherichia coli* bacteria and then converted the chemical energy from that digestion process into electricity. The microbes from the bacteria fed on the sugar, which released electrons. These electrons, in turn, supplied a charge to the battery through a redox reaction. Moreover, the sugar cubes being completely dissolved by the microbes, the system produced very little waste [60.94].

Slugbot (Fig. 60.22b) was a robotic slug catcher developed at the Bristol Robotics Laboratory that was

equipped with a long articulated arm at the end of which were located a camera used for detecting slugs and a gripper used for catching them. The robot shined red light on the ground and used the camera to identify a shiny, sluglike object, which it picked up and dropped in a hopper [60.95]. Unfortunately, funding of the project ran out before researchers succeeded in refueling the robot's batteries through slug fermentation and methane production.

Although praiseworthy, these attempts did not provide convincing evidence that they could supply enough electricity to generate useful work and they have been abandoned. However, EcoBot II, a new robot that gets closer to true energetic autonomy, is under development at the Bristol Robotics Laboratory and benefits from previous experience. EcoBot II is equipped with an array of eight MFCs, in the anodes of which bacteria found in sludge act as catalysts to generate energy from dead flies supplied by a human operator or, more precisely, from sugar contained in their exoskeleton. Later on, the robot will be made predatory, using sewage or a pheromone

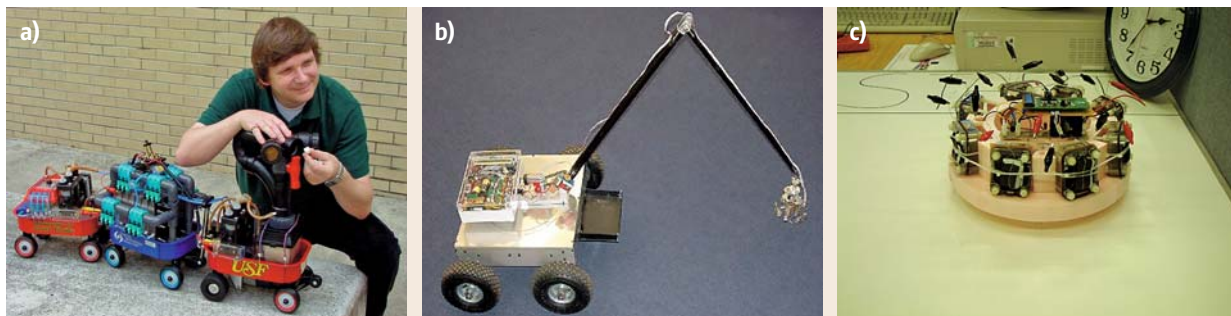


Fig. 60.22a–c Towards energetically autonomous robots: (a) Chew Chew ©Stuart Wilkinson, Mechanical Engineering Dept., University of South Florida, (b) Slugbot, and (c) EcoBot II ©Chris Melhuish, Bristol Robotics Laboratory, University of Bristol and University of the West of England

as a bait to catch the flies, and some form of pump to suck them into the digestion chambers. In the MFCs' cathodes, O_2 from free air acts as the oxidizing agent to take up the electrons and protons to produce H_2O . This closes the circuit and keeps the system balanced (Fig. 60.22c). Right now, EcoBot II can crawl along at a top speed of about 2–4 cm every 15 min and obtains

enough power to perform phototaxis while remotely reporting temperature measurements via radio at the same time [60.96]. This makes it the world's first robot to be performing four different types of behavior – sensing, processing, actuation, and communication – from unrefined biomass. These research efforts contribute to the EC project ICEA already mentioned.

60.7 Collective Robotics

Numerous research efforts contribute to the field of bio-inspired collective robotics [60.97] and several of them are described in Chap. 40, which is devoted to multiple mobile robot systems. Indeed, the collaboration of two or more workers is mandatory as soon as a given task cannot be accomplished by a single individual. This is the case, for instance, in several species of ants when workers cooperate to retrieve large preys. When one ant finds a prey item, it usually tries to move it, and, when unsuccessful for some time, recruits nestmates through direct contact or chemical marking. Within the group thus formed, ants change position and alignment until the prey can be moved toward the nest. A robotic implementation of this phenomenon is described in [60.98] and illustrates how decentralized problem solving may be implemented in a group of robots. Other demonstrations of robots that collaborate to solve a given task were produced within the framework of the SWARM-BOTS project that was funded by the EC and was focused on the design and implementation of self-organizing and self-assembling biologically inspired robots. A swarmbot is an aggregate of s-bots – i. e., a collection of mobile robots able to self-

assemble by connecting/disconnecting from each other. It can explore, navigate, and transport heavy objects on rough terrains in situations in which a single s-bot would have major problems achieving the task alone, like collectively passing over a gap too big for a single robot (Fig. 60.23) [60.99].

The coordination of a swarm of underwater glider robots in Monterey Bay is at the core of the adaptive sampling and prediction (ASAP) program, which is funded by the Office of Naval Research, and aims at measuring physicochemical parameters, and at tracking currents and upwellings [60.100]. Ultimately, the project may lead to the development of robot fleets that forecast ocean conditions and better protect endangered marine animals, track oil spills, and guide military operations at sea. Inspired by the behavior of schools of fish, the coordination policy of the robots allows them to capture the dynamic nature of the ocean while staying in organized patterns even as they are buffeted by strong currents. In particular, the paths that they follow are optimized as the ocean changes so that the measurements they take are permanently as information-rich as possible.



Fig. 60.23 A swarm-bot passing a gap ©Francesco Mondada, Project Swarm-bot, [EPFL](#)



Fig. 60.24 Mars Entomopter and refueling rover. ©NASA Institute for Advanced Concepts

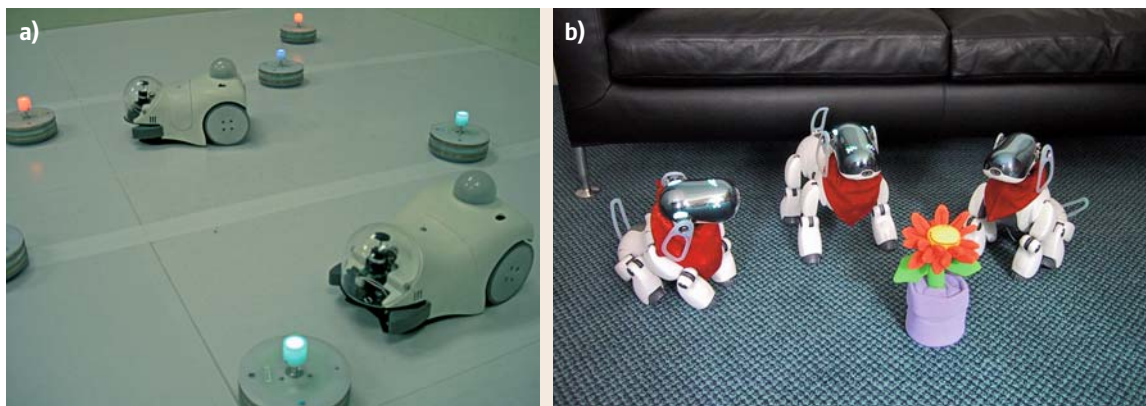


Fig. 60.25 (a) Cyber Rodents seeking battery packs ©Kenji Doya, Okinawa Institute of Science and Technology. (b) Three Sony Aibos paying attention to an object and possibly agreeing on a common word designating it ©Frédéric Kaplan, Sony Computer Science Laboratory, Paris

The NASA Institute for Advanced Concepts in Atlanta supports a project aiming at coordinating a fleet of refuelable Entomopter robots deployed from their *moth-ership*, a Pathfinder-like rover (Fig. 60.24), and flapping smartly through the thin, carbon-dioxide-laden atmosphere of Mars. With a 1 m wing span, each such robot could haul up to 15 kg of payload. A chemical muscle would generate autonomic wing beating from a liquid fuel source and provide a small amount of electricity to run onboard systems. Waste gas produced by the chemical muscle would be tapped for steering the robot while in flight, thereby making obstacle avoidance possible. Once airborne, the robots would flap at low altitude over Mars, sniffing atmospheric samples, looking for minerals, and even collecting rock and soil specimens. They could also provide the rover with essential navigation instructions. Finally, returning to home base, the Entomopters would suckle up to the rover, refueling themselves for another round of aerial maneuvers. Mini prototypes of such a flying robot have already been produced [60.101, 102] for terrestrial applications. In the same overall perspective, *Huntsberger* [60.103] describes the map-making memory and action-selection mechanism of the biologically inspired system for map-based autonomous rover control (BISMARC), an integrated control system for long-duration missions involving robots performing cooperative tasks, which is currently under development at the Jet Propulsion Laboratory (JPL) in Pasadena.

Besides cooperation, another variety of interaction is put to work in experiments that involve robots in artificial ecosystems, in which they usually compete for the acquisition of spare resources. This is, for example, the

case with the Cyber Rodent project at the Okinawa Institute of Science and Technology, a project that seeks to understand the origins of our reward and affective systems by building artificial agents that share the same intrinsic constraints as animals, i. e., self-preservation and self-reproduction. A Cyber Rodent is a robot that can search for and recharge from battery packs on the floor (Fig. 60.25a). It can also *mate* with a nearby agent, a process that entails the transfer of control programs through the robots' infrared communication ports. In particular, Cyber Rodents are used to study how evolution can help in the learning of battery-capturing behaviors, through the transfer of *genes* coding learning parameters such as the speed of memory update and the width of random exploration [60.104].

Finally, communication is at the heart of several projects that have been undertaken in the line of the so-called *talking heads* experiments [60.105] that studied the evolution of a shared lexicon in a population of embodied software agents. The agents developed their vocabulary by observing a scene through digital cameras and communicating about what they had seen together (Fig. 60.25b). Among such research efforts, the ECAgents project, which is sponsored by the EC, is developing a new generation of embodied agents that are able to interact directly with the physical world and to communicate between them and with other agents, including humans. For example, *Hafner* and *Kaplan* [60.106] studied how nonverbal communication in robots, like pointing gestures, can serve to bootstrap their shared communication systems by influencing the attention of one another. More generally, the ECAgents project investigates basic properties of

different communication systems – from simple communication systems in animals to human language and technology-supported human communication – to clarify the nature of existing communications systems and to provide ideas for designing new technologies based on collections of embodied and communicating de-

vices. This will be achieved through the development of new design principles, algorithms, and mechanisms that can extend the functionality of existing technological artefacts (mobile phone, WI-FI devices, robots and robot-like artefacts, etc.) or lead to the development of entirely new ones.

60.8 Biohybrid Robots

The solutions that nature has evolved to difficult engineering problems are, in many cases, far beyond present-day engineering capability. Therefore, when engineers are unable to reproduce the functionalities of some sensor, actuator or controller embodied in a living creature, they may try to integrate the corresponding biological component into a so-called *biohybrid* robot, thus physically using biology to augment technology. This has been done by Kuwana et al. [60.107], who equipped a mobile robot with living silk-moth antennae, the electroantennogram signals they produced being sent to an external computer that translated them into

actuator signals. In a pheromone plume, this robot exhibited a locomotion pattern similar to that of a male silk moth and succeeded in locating a pheromone source. Likewise, Herr and Dennis [60.108] built a swimming robot actuated by two explanted frog semitendinous muscles and controlled by an embedded microcontroller. The muscles got their energy from the glucose solution that the fish was swimming in. Using open-loop stimulation protocols, the robot performed basic maneuvers such as starting, stopping, turning, and straight-line swimming at a maximum speed of 1/3 body-lengths/second.

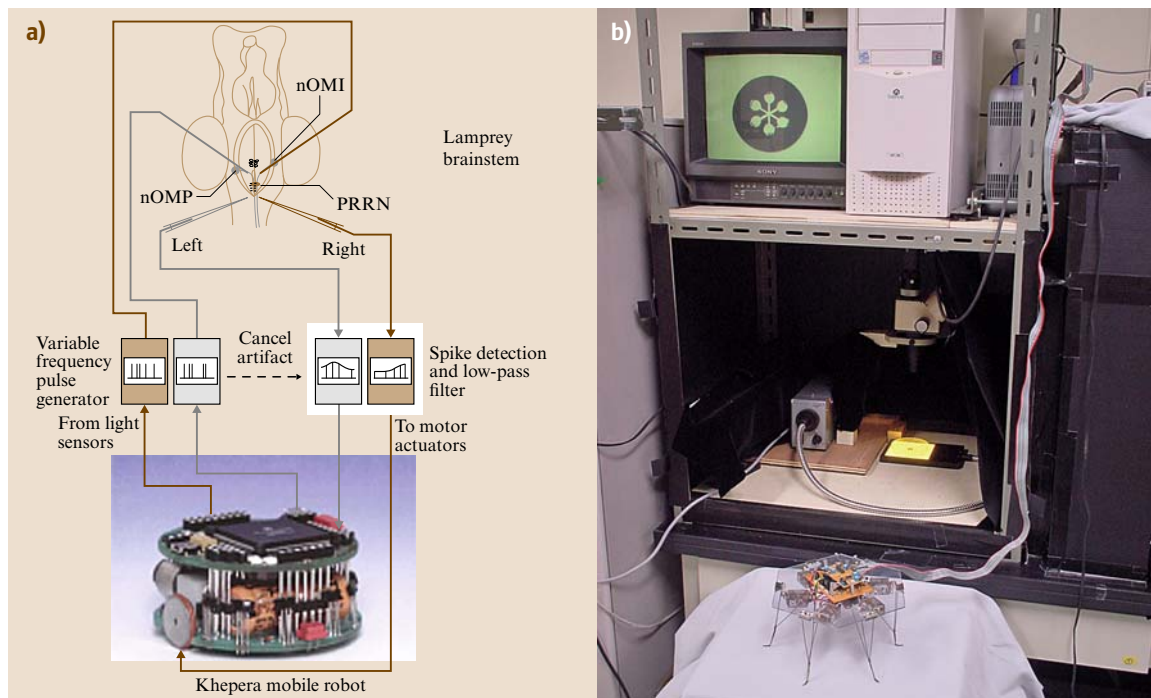


Fig. 60.26 (a) A robot controlled by a lamprey brainstem ©Sandro Mussa-Ivaldi, Depts. of Physiology, Physical Medicine & Rehabilitation and Biomedical Engineering, Northwestern University. (b) A robot controlled by a slime mold ©Soichiro Tsuda, Graduate School of Science and Technology, Kobe University

Bakkum et al. [60.109] review a series of experiences in which cultures of real neurons are used to control robots. At Northwestern University, for instance, the part of the lamprey's brain that works to keep the animal's body balanced has been connected to a two-wheeled Khepera robot. In normal circumstances, the corresponding circuit receives vestibular and other sensory signals and issues motor commands to stabilize the orientation of the body during swimming. In the experimental setup that was used, light receptors on the robot sensed the surroundings, and a computer translated that information into electrical impulses that were fed into the lamprey's neurons. The latter interpreted the impulses as they would if they were trying to keep the animal swimming upright. The computer then translated the cells' signals back into electrical commands instructing the robot how to turn its wheels in response to a light (Fig. 60.26a). Such experiments have provided useful hints about the adaptive capacities of the neuronal circuit, demonstrating that different behaviors can be generated with different electrode locations, and that the prolonged suppression of one input channel leads to altered responsiveness long after it has been restored [60.110].

A similar approach has been undertaken at the University of Southampton where, instead of calling upon numerous interconnected neurons, a single-celled organism – *Physarum polycephalum*, a bright yellow slime mold that can grow to several meters in diameter and naturally shies away from light – has been used to control the movement of a hexapod robot so that it kept out of light and sought dark places in which to hide itself. The experimenters grew the slime in a six-pointed star shape on top of a circuit and connected it remotely, via a computer, to the robot. Any light shone on sensors mounted on top of the robot was used to control light shone onto one of the six points of the circuit-mounted mold, each corresponding to a leg of the robot (Fig. 60.26b). As the slime tried to get away from the light its movement was sensed by the circuit and used to control one of the robot's six legs. The robot then scabbled away from bright lights as a mechanical embodiment of the mold [60.111].

Finally, entire brains may be used to control robotic devices, as done at Duke University, where a rhesus monkey was taught to intentionally control the movement of a robot arm in real time, using only signals from his brain and visual feedback on a video screen. A neuronal model, which was developed by monitoring normal brain and muscle activity as a monkey moved

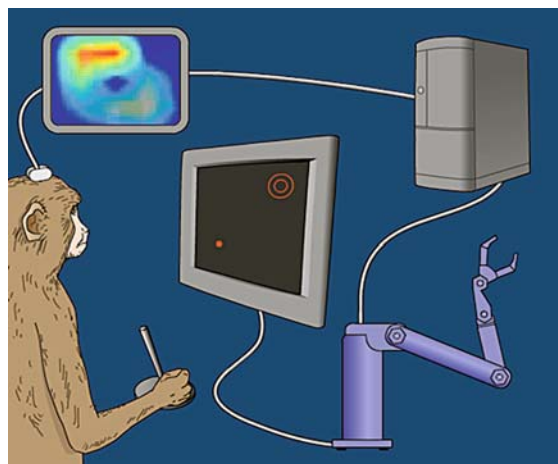


Fig. 60.27 A monkey brain controlling a robotic arm. After <http://www.biotele.com/>

his own arm, served to translate the brain signals from the monkey into movements of the robot arm. While the monkey was using a joystick to move a cursor on a computer screen, readings were taken from a few hundred neurons in the frontal and parietal regions of his brain. The activation of the biceps and wrist muscles was monitored, as were the velocity of the arms and the force of the grip. Once the neuronal model had developed an accurate level of prediction, the control of the cursor was switched from the joystick to the robotic arm, which in turn was controlled by the monkey's brain signals. At first the monkey continued moving his own arm whilst carrying out the task, but in time he learned that this was no longer necessary and stopped doing so [60.112] (Fig. 60.27).

Obviously, such technology affords great perspectives in the rehabilitation of people with brain and spinal cord damage, on the one side, while raising ethical issues, on the other side [60.113]. ETHICBOTS, an EC-sponsored project [60.114] is devoted to the identification and analysis of techno-ethical issues concerning the integration of human beings and artificial devices. Obviously, such concerns need to be extended to other animals as well, in cases where, for example, a living Madagascar hissing cockroach is placed atop a modified trackball to control a three-wheeled robot as part of an artistic project [60.115].

Finally, Chap. 33 on exoskeletons provides other examples of the integration of human beings and artificial devices, while Chap. 64 insists on the social and ethical implications of robotics.

60.9 Discussion

It appears that almost all the continuum previously evoked has been covered by the numerous realizations described in this review. In particular, it should be clear that a kangaroo-shaped toy robot has almost nothing to do with a real kangaroo, and that some of the devices that afford a robot a sense of touch or smell do reproduce a natural functionality, but certainly not in the exact way in which it is implemented in a living creature. Such realizations are clearly bio-inspired, but definitely not biomimetic, and they must be placed near the continuum's first extremity. Towards the other extremity, the optoelectronic circuits that copy the mechanisms implementing visual reflexes in the fly, or the artificial neural networks that copy the brain structures involved in navigation, action-selection and planning in mammals capitalize on almost all the relevant and currently available biological knowledge. They are clearly biomimetic even if their degree of realism may certainly be improved, if only because there are still a lot of discoveries to be made regarding the inner workings of these mechanisms and structures *in vivo*.

Noticing that important recent achievements in robotics have been inspired by living systems does not mean that purely engineering approaches to the field have not also lead to spectacular advances, as evidenced by numerous chapters in this handbook. Conversely, under the pretext that nature never invented the wheel, the jet plane or the laser range-finder, denying any usefulness to bio-inspired approaches to robotics would sound like rearguard action. In fact, microscopic wheels do exist in nature such as in Adenosine 5'-triphosphate (ATP) synthase and bacterial flagellum. Additionally, the wheel-like locomotion of tumbleweed balls across the desert has inspired the [JPL Tumbleweed rover](#), i. e., a quasi-spherical vehicle intended to traverse a planetary surface with a rolling and/or bouncing motion driven by the wind [60.116]. Instead, the interesting issue is that of delineating the applications for which such approaches are most likely to be useful to robotics. In this regard, although we repeatedly deplored several years ago [60.117, 118] that the field of robotics did not favor a profusion of comparisons that would allow us to understand which architectures and working principles allow an animat or a robot to solve a given kind of problem in a particular environment, we are compelled to observe that the situation has not improved since that time. Considering, for instance, the profusion of models, working principles, physical realizations, and the like that characterize the robotic applications to navigation [60.66, 67],

nothing would be more useful than systematic comparisons in which different robots would be confronted with the same problem, and different problems would be tackled by the same robot.

Be that as it may, the main lesson to be drawn from this review is probably that, if human inventions may be irreplaceable for optimizing a given functionality in rather predictable circumstances, drawing inspiration from the solutions discovered via natural tinkering [60.119] may be particularly useful for finding operational compromises to multi-optimization problems raised by survival issues in unpredictable environments, i. e., to issues that engineers have carefully postponed as much as possible up to now. Indeed, probably few people would deny that the capacities of autonomy and adaptation exhibited by living creatures far exceed those of current robots, which are seldom confronted with the necessity of coping with permanent changes in their external environment or in their inner needs, motivations or emotions, nor at the constraint of freeing themselves from human-delivered energy resources.

Besides the fact that numerous sensors, actuators or control architectures in animals are often still more efficient than the artificial devices they have inspired – either for reasons tied to technological limitations or to lack of biological knowledge – perhaps the principal reason for the superiority of animals over robots lies in their greater degree of integration. In fact, in the 3.5 billion years since the appearance of life on Earth, natural sensors, effectors, and control architectures have been offered enough time to coevolve and produce coherent wholes, a process that contrasts strongly with the current practice of engineers, who often independently design and produce the various components that they later assemble into a given artifact. Unfortunately, the laws governing natural evolution and integration are far from being deciphered and exploited in a more efficient manner than in current evolutionary robotics applications.

This last remark is related to the observation that biologists too, in their tendency to favor reductionist over holistic approaches, often postpone the consideration of integrative mechanisms and processes that future robotics will mostly need. Indeed, in its endeavor to unravel the mysteries of natural life, traditional biology usually seeks to decompose a system into its constituent subsystems and then to study these in isolation from one another, according to a top-down, analytical, and reductionist approach. On the contrary, people involved in

artificial life or robotics research attempt to reproduce given characteristics of living systems with man-made artifacts such as computers or robots. Ideally, their approach is bottom-up and starts with a collection of entities or modules exhibiting properties or behaviors that are simple and well understood, and organizes them into more complex systems in which internal interac-

tions generate emergent lifelike properties. Obviously, fruitful interactions are to be expected between these people and biologists who will devote their research efforts to the kind of integrative considerations advocated here. Many such interactions are already in place, as demonstrated in this chapter and several others in this handbook.

60.10 Conclusion

This chapter has reviewed numerous recent applications of bio-inspired solutions to robotics. It seems likely that such solutions will prove to be even more useful as future robots are confronted with similar survival issues to those experienced by animals in unpredictable environments. This will require subsequent progress in the corresponding biological knowledge, a process to which tight collaboration between numerous disciplines, including robotics, may well critically contribute.

Further Reading

Interested readers may find useful additional information in the following textbooks:

- J. Ayers, J. Davis, A. Rudolph: *Neurotechnology for Biomimetic Robots* (MIT Press, Cambridge 2002)
- Y. Bar-Cohen: *Biomimetics – Biologically Inspired Technologies* (CRC, Boca Raton 2005)
- Y. Bar-Cohen, C. Breazeal: *Biologically Inspired Intelligent Robots* (SPIE, Bellingham 2003)
- R. Duro, J. Santos, M. Graña, J. Kacprzyk: *Biologically Inspired Robot Behavior Engineering* (Springer, Berlin, Heidelberg 2003)
- B. Webb, T.R. Consi: *Biorobotics. Methods and Applications* (MIT Press, Cambridge 2001)

References

- 60.1 R. Cordeschi: *The Discovery of the Artificial: Behavior, Mind and Machines Before and Beyond Cybernetics* (Kluwer, Dordrecht 2002)
- 60.2 L. Paulson: Biomimetic robots, *Computer* **37**(9), 48–53 (2004)
- 60.3 C. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2002)
- 60.4 K. Itoh, H. Miwa, M. Zecca, H. Takanobu, S. Roccella, M. Carozza, P. Dario, A. Takanishi: Mechanical design and motion control of emotion expression humanoid robot WE-4R, *Proc. 16-th CISM-IFTOMM Symp. Robot Des. Dyn. Contr.* (2006) pp.255–262
- 60.5 K. MacDorman, H. Ishiguro: The uncanny advantage of using androids in cognitive and social science research, *Interact. Stud.* **7**(3), 297–337 (2006)
- 60.6 W. Shen, B. Salemi, P. Will: Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots, *IEEE Trans. Robot. Autom.* **18**(5), 700–712 (2002)
- 60.7 W. Shen: Polymorphic Robotics Laboratory, University of Southern California, Marina del Ray, <http://www.isi.edu/robots/>
- 60.8 N. Franceschini, J.M. Pichon, C. Blanes: From insect vision to robot vision, *Philos. Trans. R. Soc. London B* **337**, 283–294 (1992)
- 60.9 S. Viollet, N. Franceschini: Aerial minirobot that stabilizes and tracks with a bio-inspired visual scanning sensor. In: *Biorobotics. Methods and Applications*, ed. by B. Webb, T. Consi (MIT Press, Cambridge 2001) pp. 67–83
- 60.10 F. Ruffier, N. Franceschini: Optic flow regulation: the key to aircraft automatic guidance, *Robot. Auton. Syst.* **50**(4), 177–194 (2005)
- 60.11 R. Möller, D. Lambrinos, R. Pfeifer, T. Labhart, R. Wehner: Modeling ant navigation with an autonomous agent, *From Animals to Animats 5. Proc. 5-th Int. Conf. Simul. Adapt. Behav.* (MIT Press, Cambridge 1998) pp.185–194
- 60.12 A. Ude, C. Atkeson, G. Cheng: Combining peripheral and foveal humanoid vision to detect, pursue, recognize and act, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.* (Las Vegas 2003) pp. 2173–2178
- 60.13 G. Metta: An attentional system for a humanoid robot exploiting space variant vision, *IEEE-RAS Int. Conf. Humanoid Robots*, Vol.22–24 (Tokyo 2001) pp. 359–366

- 60.14 A. Davison, O. Stasse, K. Yokoi: Vision based SLAM for a humanoid robot, Int. Conf. Robot. Autom. (ICRA), SLAM Workshop, Vol. 16 (2005)
- 60.15 P. Michel, J. Chestnutt, J. Kuffner, T. Kanade: Vision-guided humanoid footstep planning for dynamic environments, Proc. IEEE-RAS Conf. Humanoid Robots (Humanoids) (2005) pp.13–18
- 60.16 R. Reeve, A. van Schaik, C. Jin, T. Hamilton, B. Torben-Neilsen, B. Webb: Directional hearing in a silicon cricket, Biosystems **87**(2–3), 307–313 (2007)
- 60.17 R. Reeve, B. Webb: New neural circuits for robot phonotaxis, Philos. Trans. R. Soc. London A **361**, 2245–2266 (2003)
- 60.18 R. Müller, J. Hallam: Knowledge mining for biomimetic smart antenna shapes, Robot. Auton. Syst. **50**, 131–145 (2005)
- 60.19 R. Kuc: Biomimetic sonar recognizes objects using binaural information, J. Acoust. Soc. Am. **102**(2), 689–696 (1997)
- 60.20 K. Nakadai, H. Okuno, H. Kitano: Robot recognizes three simultaneous speech by active audition, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2003) pp. 398–405
- 60.21 N. Cowan, E. Ma, M. Cutkosky, R. Full: A biologically inspired passive antenna for steering control of a running robot, Springer Tracts Adv. Robot. **15**, 541–550 (2005)
- 60.22 R. Russell, J. Wijaya: Object location and recognition using whisker sensors, Aust. Conf. Robot. Autom. (2003), CD-ROM Proceedings ISBN 0-9587583-5-2
- 60.23 M.J. Pearson, I. Gilhespy, C. Melhuish, B. Mitchinson, M. Nibouche, A.G. Pipe, T.J. Prescott: A biomimetic haptic sensor, Int. J. Adv. Robot. Syst. **2**(4), 335–343 (2005)
- 60.24 T. Someya, Y. Kato, T. Sekitani, S. Iba, Y. Noguchi, Y. Murase, H. Kawaguchi, T. Sakurai: Conformable, flexible, large-area networks of pressure and thermal sensors with organic transistor active matrixes, PNAS **102**, 12321–12325 (2005)
- 60.25 V. Maheshwari, R. Saraf: High-resolution thin-film device to sense texture by touch, Science **312**(5779), 1501–1504 (2006)
- 60.26 T. Morse, T. Ferrée, S. Lockery: Robust spatial navigation in a robot inspired by chemotaxis in *Caenorhabditis elegans*, Adapt. Behav. **6**(3–4), 393–410 (1998)
- 60.27 F. Grasso: Environmental information, animal behavior, and biorobot design. Reflections on locating chemical sources in marine environments. In: *Biorobotics. Methods and Applications*, ed. by B. Webb, T.R. Consi (MIT Press, Cambridge 2001) pp. 21–35
- 60.28 K. Itoh, H. Miwa, M. Matsumoto, M. Zecca, H. Takanobu, S. Roccella, M.C. Carrozza, P. Dario, A. Takanishi: Various emotional expressions with emotion expression humanoid robot WE-4RIL, Proc. 1-st IEEE Tech. Exhib. Based Conf. Robot. Autom. (2004) pp. 35–36
- 60.29 A.J. Ijspeert, A. Crespi, D. Ryczko, J.M. Cabelguen: From swimming to walking with a salamander robot driven by a spinal cord model, Science **315**(5817), 1416–1420 (2007)
- 60.30 D. Tsakiris, M. Sfakiotakis, A. Menciassi, G. LaSpina, P. Dario: Polychaete-like undulatory robotic locomotion, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2005) pp. 3029–3034
- 60.31 J. Ayers, J. Witting: Biomimetic approaches to the control of underwater walking machines, Philos. Trans. A. Math. Phys. Eng. Sci. **365**(1850), 273–295 (2007)
- 60.32 J. Clark, J. Cham, S. Bailey, E. Froehlich, P. Nahata, R. Full, M. Cutkosky: Biomimetic design and fabrication of a hexapedal running robot, Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (2001) pp. 3643–3649
- 60.33 M.C. Birch, R.D. Quinn, G. Hahm, S.M. Phillips, B.T. Drennan, A.J. Fife, R.D. Beer, X. Yu, G. Garverick, S. Laksanacharoen, A.J. Pollack, R.E. Ritzmann: Miniature hybrid robot propelled by legs, IEEE Robot. Autom. Mag. **10**, 20–30 (2002)
- 60.34 Y. Song, M. Sitti: Surface tension driven biologically inspired water strider robots: theory and experiments, IEEE Trans. Robot. **23**(3), 578–589 (2007)
- 60.35 I. Poulakakis, J. Smith, M. Buehler: Modeling and experiments of untethered quadrupedal running with a bounding gait: The Scout II robot, Int. J. Robot. Res. **24**(4), 239–256 (2005)
- 60.36 T. Geng, B. Porr, F. Wörgötter: Fast biped walking with a sensor-driven neuronal controller and real-time online learning, Int. J. Robot. Res. **25**(3), 243–259 (2006)
- 60.37 K. Sangbae, A. Asbeck, M. Cutkosky, W. Provancher: SpinybotII: climbing hard walls with compliant microspines, Proc. 12-th Int. Conf. Adv. Robot. (ICAR) (2005) pp. 601–606
- 60.38 A.T. Asbeck, K. Sangbae, M.R. Cutkosky, W.R. Provancher, M. Lanzetta: Scaling hard vertical surfaces with compliant microspine arrays, Int. J. Robot. Res. **25**(12), 1165–1179 (2006)
- 60.39 M. Spenko, M. Cutkosky, C. Majidi, R. Fearing, R. Groff, K. Autumn: Foot design and integration for bioinspired climbing robots, Proc. SPIE Unmanned Systems Technology VIII, Vol. 6230–19 (2006) pp. 1–12
- 60.40 U. Scarfogliero, C. Stefanini, P. Dario: A bioinspired concept for high efficiency locomotion in micro robots: the jumping Robot Grillo, Proc. IEEE Int. Conf. Robot. Autom. (2006) pp. 4037–4042
- 60.41 D. Barrett, M. Triantafyllou, D. Yue, M. Grosenbaugh, M. Wolfgang: Drag reduction in fish-like locomotion, J. Fluid Mech. **392**, 183–212 (1999)
- 60.42 F. Fish: Limits of nature and advances of technology: What does biomimetics have to offer to aquatic robots?, Appl. Bion. Biomech. **3**(1), 49–60 (2006)

- 60.43 J.H. Long, J. Schumacher, N. Livingston, M. Kemp: Four flippers or two? Tetrapodal swimming with an aquatic robot, *Bioinsp. Biomimet* **1**, 20–29 (2006)
- 60.44 X. Deng, L. Schenato, W.C. Wu, S. Sastry: Flapping flight for biomimetic robotic insects: Part I—System modeling, *IEEE Trans. Robot.* **22**(4), 776–788 (2006)
- 60.45 X. Deng, L. Schenato, W.C. Wu, S. Sastry: Flapping flight for biomimetic robotic insects: Part II—Flight control design, *IEEE Trans. Robot.* **22**(4), 789–803 (2006)
- 60.46 P. Zdunich, D. Bilyk, M. MacMaster, D. Loewen, J. DeLaurier, R. Kornbluh, T. Low, S. Stanford, D. Holeman: The development and testing of the Mentor flapping-wing micro air vehicle, *J. Aircraft* (in press)
- 60.47 T. Mueller, J. DeLaurier: An overview of micro air vehicle aerodynamics. In: *Fixed and Flapping Wing Aerodynamics for Micro Air Vehicle Applications*, Progress in Astronautics and Aeronautics, Vol. 195, ed. by T. Mueller (AIAA, Reston 2002) pp. 1–9
- 60.48 K. Jones, C. Bradshaw, J. Papadopoulos, M. Platzer: Development and flight testing of flapping-wing propelled micro air vehicles, 2nd AIAA "Unmanned Unlimited" Conf. Workshop Exhibit, Vol. 6549 (AIAA, San Diego 2003)
- 60.49 K. Jones, C. Bradshaw, J. Papadopoulos, M. Platzer: Improved performance and control of flapping-wing propelled micro air vehicles, 42nd AIAA Aerospace Sci. Meeting Exhibit, Vol. 399 (AIAA, Reno 2004)
- 60.50 N. Chronister: The Ornithopter Zone, Rochester, <http://www.ornithopter.org/flappingwings/toporov2.htm> (last accessed December 2007)
- 60.51 J. DeLaurier: Institute for Aerospace Studies, University of Toronto, Canada, http://www.ornithopter.ca/index_e.html (last accessed December 2007)
- 60.52 J. DeLaurier: The development and testing of a full-scale piloted ornithopter, *Cdn. Aeronaut. Space J.* **45**(2), 72–82 (1999)
- 60.53 I. Walker, D. Dawson, T. Flash, F. Grasso, R. Hanlon, B. Hochner, W. Kier, M. Pagano, C. Rahn, Q. Zhang: Continuum robot arms inspired by cephalopods, *Proc. SPIE*, Vol. 5804 (2005) pp. 303–314
- 60.54 I. Walker, C. Carreras, R. McDonnell, G. Grimes: Extension versus bending for continuum robots, *Int. J. Adv. Robot. Syst.* **3**(2), 171–178 (2006)
- 60.55 P. Scarfe, E. Lindsay: Air muscle actuated low cost humanoid hand, *Int. J. Adv. Robot. Syst.* **3**(1), 139–146 (2006)
- 60.56 M. Carrozza, P. Dario, F. Vecchi, S. Roccella, M. Zecca, F. Sebastiani: The CYBERHAND: on the design of a cybernetic prosthetic hand intended to be interfaced to the peripheral nervous system, *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (2003) pp. 2642–2647
- 60.57 Y. Gao, A. Ellery, M. Jaddou, J. Vincent, S. Eckersley: A novel penetration system for in situ astrobiological studies, *Int. J. Adv. Robot. Syst.* **2**(4), 281–286 (2005)
- 60.58 D. Dennett: Why not the whole iguana?, *Behav. Brain Sci.* **1**, 103–104 (1978)
- 60.59 R. Brooks: A robust layered control system for a mobile robot, *IEEE Robot. Autom.* **2**(1), 14–23 (1986)
- 60.60 R. Brooks: *Cambrian Intelligence. The Early History of the New AI* (MIT Press, Cambridge 1999)
- 60.61 J. Connell: *Minimalist Mobile Robotics. A Colony-Style Architecture for an Artificial Creature* (Academic, New York 1990)
- 60.62 R. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)
- 60.63 J. Piaget: *Biology and Knowledge: An Essay on the Relations between Organic Relations and Cognitive Processes* (Univ. Chicago Press, Chicago 1971)
- 60.64 M. Arbib: Schema theory. In: *The Handbook of Brain Theory and Neural Networks*, ed. by M. Arbib (MIT Press, Cambridge 1995)
- 60.65 F. Saito, T. Fukuda: A first result of the Brachiator III – A new brachiation robot modeled on a siamang, *Proc. ALife V*, ed. by C. Langton, K. Shimohara (MIT Press, Cambridge 1996) pp. 354–360
- 60.66 D. Filliat, J.A. Meyer: Map-based navigation in mobile robots – I. A review of localization strategies, *J. Cognit. Syst. Res.* **4**(4), 243–282 (2003)
- 60.67 J.A. Meyer, D. Filliat: Map-based navigation in mobile robots – II. A review of map-learning and path-planning strategies, *J. Cogn. Syst. Res.* **4**(4), 283–317 (2003)
- 60.68 O. Trullier, S. Wiener, A. Berthoz, J.A. Meyer: Biologically-based artificial navigation systems: Review and prospects, *Prog. Neurobiol.* **51**, 483–544 (1997)
- 60.69 M. Khamassi, L. Lachèze, B. Girard, A. Berthoz, A. Guillot: Actor-critic models of reinforcement learning in the basal ganglia: From natural to artificial rats, *Adapt. Behav.* **13**(2), 131–148 (2005), Special issue towards artificial rodents
- 60.70 F. Montes-Gonzalez, T. Prescott, K. Gurney, M. Humphries, P. Redgrave: An embodied model of action selection mechanisms in the vertebrate brain, From Animals to Animats 6. *Proc. 6-th Int. Conf. Simul. Adapt. Behav.* (MIT Press, Cambridge 2000) pp. 157–166
- 60.71 B. Girard, D. Filliat, J.A. Meyer, A. Berthoz, A. Guillot: Integration of navigation and action selection functionalities in a computational model of cortico-basal ganglia-thalamo-cortical loops, *Adapt. Behav.* **13**(2), 115–130 (2005)
- 60.72 J.A. Meyer, A. Guillot, B. Girard, M. Khamassi, P. Pirim, A. Berthoz: The Psikharpa project: Towards building an artificial rat, *Robot. Auton. Syst.* **50**(4), 211–223 (2005)
- 60.73 <http://www.his.sunderland.ac.uk/mirrorbot/> (last accessed December 2007)
- 60.74 G. Rizzolatti, L. Craighero: The mirror-neuron system, *Annu. Rev. Neurosci.* **27**, 169–192 (2004)

- 60.75 S. Wermter, G. Palm, M. Elshaw (Eds.): *Biomimetic Neural Learning for Intelligent Robots – Intelligent Systems, Cognitive Robotics, and Neuroscience*, Lect. Notes Comput. Sci., Vol. 3575 (Springer, Berlin 2005)
- 60.76 J. Krichmar, G. Edelman: Brain-based devices: Intelligent systems based on principles of the nervous system, Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Vol. 1 (2003) pp. 940–945
- 60.77 A. Seth, J. McKinstry, G. Edelman, J. Krichmar: Spatiotemporal processing of whisker input supports texture discrimination by a brain-based device, From Animals to Animats 8. Proc. 8-th Int. Conf. Simul. Adapt. Behav. (MIT Press, Cambridge 2004) pp. 130–139
- 60.78 J. Krichmar, A. Seth, D. Nitz, J. Fleischer, G. Edelman: Spatial navigation and causal analysis in a brain-based device modeling cortical-hippocampal interactions, Neuroinformatics 3(3), 197–222 (2005)
- 60.79 R. Kozma, D. Wong, M. Demirer, W. Freeman: Learning intentional behavior in the K-model of the amygdala and entorhinal cortex with the cortico-hippocampal formation, Neurocomputing 65–66, 23–30 (2005)
- 60.80 S. Nolfi, D. Floreano: *Evolutionary Robotics. The Biology, Intelligence, and Technology of Self-Organizing Machines* (MIT Press, Cambridge 2000)
- 60.81 W. Langdon, P. Nordin: Evolving hand-eye coordination for a humanoid robot with machine code genetic programming, Lect. Notes Comput. Sci. 2038, 313–324 (2001)
- 60.82 D. Filliat, J. Kodjabachian, J.A. Meyer: Evolution of neural controllers for locomotion and obstacle-avoidance in a 6-legged robot, Connect. Sci. 11, 223–240 (1999)
- 60.83 R. Barate, S. Doncieux, J.A. Meyer: Design of a bio-inspired controller for dynamic soaring in a simulated UAV, Bioinsp. Biomimet. 1, 76–88 (2006)
- 60.84 S. Doncieux, J.B. Mouret, A. Angeli, R. Barate, J.A. Meyer, E. de Margerie: Building an artificial bird: Goals and accomplishments of the Robur project, Proc. Eur. Micro Aerial Vehicles Conf. (EMAV, Braunschweig 2006)
- 60.85 H. Lipson, J. Pollack: Automatic design and manufacture of robotic lifeforms, Nature 406, 974–978 (2000)
- 60.86 J. Kodjabachian, J.A. Meyer: Evolution and development of control architecture in animats, Robot. Auton. Syst. 16, 161–182 (1995)
- 60.87 J.A. Meyer, S. Doncieux, D. Filliat, A. Guillot: Evolutionary approaches to neural control of rolling, walking, swimming and flying animats or robots. In: *Biologically inspired robot behavior engineering*, ed. by R. Duro, J. Santos, M. Grana (Springer, Berlin, Heidelberg 2003) pp. 1–43
- 60.88 M. Lungarella, G. Metta: Beyond gazing, pointing, and reaching: A survey of developmental robotics, Proc. 3-rd Int. Workshop Epigenet. Robot. Modeling Cognitive Development in Robotic Systems (2003) pp. 81–89
- 60.89 J. Piaget: *The Child's Conception of the World* (LITTLEFIELD ADAMS, New York 1990)
- 60.90 R. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati, M. Williamson: The Cog project: Building a humanoid robot. In: *Computation for Metaphors, Analogy, and Agents*, Vol. LNCS 1562, ed. by C. Nehaniv (Springer, Berlin, Heidelberg) pp. 52–87
- 60.91 R. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati, M. Williamson: The Cog project: Building a humanoid robot, Lect. Notes Comput. Sci. 1562, 52–87 (1999)
- 60.92 B. Adams, C. Breazeal, R. Brooks, B. Scassellati: Humanoid robots: A new kind of tool, IEEE Intell. Syst. Appl. 15(4), 25–31 (2000)
- 60.93 P.Y. Oudeyer, F. Kaplan, V. Hafner, A. Whyte: The playground experiment: Task-independent development of a curious robot, Proc. AAAI Spring Symp. Devel. Robot., ed. by D. Bank, L. Meeden (2005) pp. 42–47
- 60.94 S. Wilkinson: Gastrobots – Benefits and challenges of microbial fuel cells in food powered robot applications, Auton. Robots 9, 99–111 (2000)
- 60.95 I. Kelly: The design of a robotic predator: The Slug-Bot, Robotica 21, 399–406 (2003)
- 60.96 I. Ieropoulos, C. Melhuish, J. Greenman, I. Horsfield: EcoBot-II: An artificial agent with a natural metabolism, Int. J. Adv. Robot. Syst. 2(4), 295–300 (2005)
- 60.97 C. Kube, C. Parker, T. Wang, H. Zhang: Biologically inspired collective robotics. In: *Recent Developments in Biologically Inspired Computing*, ed. by L. DeCastro, F. von Zuben (Idea, New York 2004) pp. 367–396
- 60.98 C. Kube, E. Bonabeau: Cooperative transport by ants and robots, Robot. Auton. Syst. 30(1), 85–101 (2000)
- 60.99 F. Mondada, M. Bonani, S. Magnenat, A. Guignard, D. Floreano: Physical connections and cooperation in swarm robotics, Proc. 8-th Conf. Intell. Auton. Syst. (IAS8) (2004) pp. 53–60
- 60.100 D. Paley, F. Zhang, N. Leonard: Cooperative Control for Ocean Sampling: The Glider Coordinated Control System, IEEE Trans. Contr. Syst. Techn. In press.
- 60.101 R.C. Michelson: The Entomopter. In: *Neurotechnology for Biomimetic Robots*, ed. by J. Ayers, J. Davis, A. Rudolph (MIT Press, Cambridge 2002) pp. 481–509
- 60.102 R.C. Michelson: Novel approaches to miniature flight platforms, J. Aerosp. Eng. 218, 363–373 (2004)
- 60.103 T. Huntsberger: Biologically inspired autonomous rover control, Auton. Robots 8, 1–6 (2001)
- 60.104 K. Doya, E. Uribe: The Cyber Rodent Project: Exploration of adaptive mechanisms for self-preservation and self-reproduction, Adapt. Behav. 13(2), 149–160 (2005)

- 60.105 L. Steels: Evolving grounded communication for robots, *Trends Cogn. Sci.* **7**(7), 308–312 (2003)
- 60.106 V. Hafner, F. Kaplan: Learning to interpret pointing gestures: experiments with four-legged autonomous robots, *Lect. Notes Artif. Intell.* **3575**, 225–234 (2005)
- 60.107 Y. Kuwana, I. Shimoyama, Y. Sayama, H. Miura: Synthesis of pheromone-oriented emergent behavior of a silkworm moth, *Proc. IROS*, Vol. 96 (1996) pp. 1722–1729
- 60.108 H. Herr, B. Dennis: A swimming robot actuated by living muscle tissue, *J. Neuroeng. Rehabil.* **1**(6), 1–9 (2004)
- 60.109 D. Bakkum, A. Shkolnik, G. Ben-Ary, P. Gamblen, T. DeMarse, S. Potter: Removing some 'A' from AI: Embodied cultured networks, *Lect. Notes Comput. Sci.* **3139**, 130–145 (2004)
- 60.110 B. Reger, K. Fleming, V. Sanguineti, S. Alford, F. Mussa-Ivaldi: Connecting brains to robots: an artificial body for studying the computational properties of neural tissue, *Artif. Life* **6**, 307–324 (2000)
- 60.111 S. Tsuda, K. Zauner, Y. Gunji: Robot control: from silicon circuitry to cells, *Biol. Inspired Approaches Adv. Inform. Tech. (BioADIT)* (2006) pp. 20–32
- 60.112 M. Lebedev, J. Carmena, J. O'Doherty, M. Zacksenhouse, C. Henriquez, J. Principe, M. Nicolelis: Cortical ensemble adaptation to represent velocity of an artificial actuator controlled by a brain-machine interface, *J. Neurosci.* **25**(19), 4681–4693 (2005)
- 60.113 Frost & Sullivan: *Report on Emerging Trends and Innovations in Biomechatronics* (Technical Insights Inc., Farmington 2006)
- 60.114 G. Tamburrini: Università di Napoli Federico II, Naples, <http://ethicbots.na.infn.it/> (last accessed December 2007)
- 60.115 G. Hertz: New Media Arts Lab, California Institute for Telecommunications and Information Technology, University of California, Irvine, <http://www.conceptlab.com/roachbot/> (last accessed December 2007)
- 60.116 A. Behar, J. Matthews, F. Carsey, J. Jones: NASA/JPL Tumbleweed Polar Rover, *IEEE Aerosp. Conf. Proc.*, Vol. 1 (2004) pp. 395–395
- 60.117 J.A. Meyer, A. Guillot: From SAB90 to SAB94: Four years of animat research, *From Animals to Animats 3. Proc. 3-rd Int. Conf. Simul. Adapt. Behav.* (MIT Press, Cambridge 1994)
- 60.118 A. Guillot, J.A. Meyer: From SAB94 to SAB2000: What's new, animat?, *From Animals to Animats 6. Proc. 6-th Int. Conf. Simul. Adapt. Behav.* (MIT Press, Cambridge 2000)
- 60.119 F. Jacob: Evolution and tinkering, *Science* **196**(4295), 1161–1166 (1977)

Evolutionary

61. Evolutionary Robotics

Dario Floreano, Phil Husbands, Stefano Nolfi

Evolutionary Robotics is a method for automatically generating artificial brains and morphologies of autonomous robots. This approach is useful both for investigating the design space of robotic applications and for testing scientific hypotheses of biological mechanisms and processes. In this chapter we provide an overview of methods and results of Evolutionary Robotics with robots of different shapes, dimensions, and operation features. We consider both simulated and physical robots with special consideration to the transfer between the two worlds.

61.1	Method	1423
61.2	First Steps	1424
	61.2.1 Evolution of Neural Controllers for Walking	1426
61.3	Simulation and Reality	1428
61.4	Simple Controllers, Complex Behaviors ...	1429

61.5	Seeing the Light	1431
	61.5.1 Coevolution of Active Vision and Feature Selection	1432
61.6	Computational Neuroethology	1434
	61.6.1 Emergence of Place Cells	1435
	61.6.2 Spiking Neurons.....	1436
	61.6.3 GasNets	1438
61.7	Evolution and Learning	1439
	61.7.1 Learning to Adapt to Fast Environmental Variations....	1440
	61.7.2 Evolution of Learning	1441
61.8	Competition and Cooperation	1442
	61.8.1 Coevolving Predator and Prey Robots.....	1442
	61.8.2 Evolving Cooperative Behavior	1443
61.9	Evolutionary Hardware	1444
	61.9.1 Evolvable Hardware Robot Controllers.....	1444
	61.9.2 Evolving Bodies.....	1445
61.10	Closing Remarks	1447
	References	1447

61.1 Method

Evolutionary robotics is a method for the automatic creation of autonomous robots [61.1]. It is inspired by the Darwinian principle of selective reproduction of the fittest, captured by evolutionary algorithms [61.2]. In evolutionary robotics, robots are considered as autonomous artificial organisms that develop their own control system and body configuration in close interaction with the environment without human intervention. Drawing inspiration from principles of biological self-organization, evolutionary robotics includes elements of evolutionary, neural, developmental, and morphological systems. The idea that an evolutionary process could drive the generation of control systems dates back to

at least the 1950s [61.3] with a more explicit form appearing in the mid 1980s with the ingenious thought experiments by neuroscientist *Valentino Braitenberg* on neurally driven vehicles [61.4]. In the early 1990s, the first generation of simulated artificial organisms with a genetic code describing the neural circuitry and morphology of a sensory motor system began evolving on computer screens [61.5–8]. At that time, real robots were still complicated and expensive machines that required specialized programming techniques and skillful manipulation. Towards the end of that period, a new generation of robots started to emerge that shared important characteristics with simple biological systems:

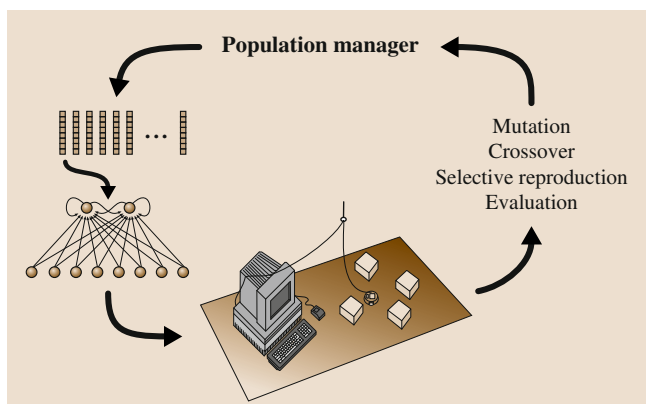


Fig. 61.1 Evolutionary experiments on a single robot. Each individual of the population is decoded into a corresponding neurocontroller which reads sensory information and sends motor commands to the robot every 300 ms while its fitness is automatically evaluated and stored away for reproductive selection

robustness, simplicity, small size, flexibility, and modularity [61.9, 10]. Above all, those robots were designed so that they could be programmed and manipulated by people without engineering training. Those technological achievements, together with the growing influence of biological inspiration in artificial intelligence [61.11], coincided with the first evolutionary experiments on real robots [61.12–14], and the term evolutionary robotics was coined [61.15].

61.2 First Steps

In an early experiment on robot evolution without human intervention, carried out at Ecole Polytechnique Fédérale de Lausanne (EPFL) [61.12], a small wheeled robot was evolved for navigation in a looping maze (Fig. 61.2). The Khepera robot has a diameter of 55 mm and two wheels with controllable velocities in both directions of rotation. It also has eight infrared sensors, six on one side and two on the other side, that can function either in active mode to measure distance from obstacles or in passive mode to measure the amount of (infrared) light in the environment. The robot was connected to a desktop computer through rotating contacts that provided both power supply and data exchange through a serial port.

A simple genetic algorithm [61.16] was used to evolve the synaptic strengths of a neural network composed of eight sensory neurons and two motor neurons.

The major methodological steps in evolutionary robotics proceed as follows (Fig. 61.1). An initial population of different artificial chromosomes, each encoding the control system (and possibly the morphology) of a robot, is randomly created. Each of these chromosomes is then decoded into a corresponding controller, for example a neural network, and downloaded into the processor of the robot. The robot is then let free to act (move, look around, manipulate the environment) according to a genetically specified controller while its performance for a given task is automatically evaluated. Performance evaluation is done by a fitness function that measures how fast and straight the robot moves, how frequently it collides with obstacles, etc. This procedure is repeated for all chromosomes of the population. The fittest individuals (those that have received more fitness points) are allowed to reproduce by generating copies of their chromosomes with the addition of random modifications introduced by genetic operators (e.g., mutations and exchange of genetic material). The newly obtained population is then tested again on the same robot. This process is repeated for a number of generations until an individual is born which satisfies the fitness function set by the user. The control system of evolved robots, encoded in an artificial genome, is therefore generated by a repeated process of selective reproduction, random mutation, and genetic recombination, similarly to what happens in natural evolution.

Each sensory unit was clamped to one of the eight active infrared sensors whose value was updated every 300 ms. Each motor unit received weighted signals from the sensory units and from the other motor unit, plus a recurrent connection with itself with a 300 ms delay. The net input of the motor units was offset by a modifiable threshold and passed through a logistic squashing function. The resulting outputs, in the range [0, 1], were used to control the two motors so that an output of 1 generated maximum rotation speed in one direction, an output of 0 generated maximum rotation speed in the opposite direction, and an output of 0.5 did not generate any motion in the corresponding wheel. A population of 80 individuals, each coding the synaptic strengths and threshold values of the neural controllers, was initialized with all weights set to small random values centered around zero. Each individual was tested on the phys-

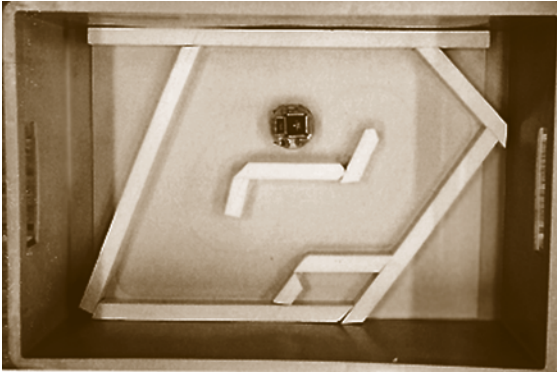


Fig. 61.2 Bird's-eye view of the desktop Khepera robot in the looping maze

ical robot for 80 sensorimotor cycles (approximately 24 s) and evaluated at every cycle according to a fitness function with three components measured onboard the robot:

$$\phi = V(1 - \sqrt{\Delta v})(1 - i), \quad (61.1)$$

where V is the average rotation speed of the two wheels, Δv is the absolute value of the algebraic difference between the signed speed values of the wheels (positive is one direction, negative the other), and i is the normalized activation value of the infrared sensor with the highest activity. The first component is maximized by speed, the second by straight motion, and the third by distance from objects.

During the first 100 generations, both average and best fitness values grew steadily, as shown in Fig. 61.3. A fitness value of 1.0 would correspond to a robot moving straight at maximum speed in an open space and therefore was not attainable in the looping maze shown in Fig. 61.2, where some of the sensors were often active and where several turns were necessary to navigate. Figure 61.4 shows the trajectory of the best individual of the last generation.

Although the fitness function did not specify in what direction the robot should navigate (given that it was perfectly circular and that the wheels could rotate in both directions), after a few generations all the best individuals moved in the direction corresponding to the side with the highest number of sensors. Individuals moving in the other direction had a higher probability of colliding into corners without detecting them and thus disappeared from the population. Furthermore, the cruising speed of the best evolved robots was approximately half of the maximum speed that could be technically achieved and did not increase even when the evolution-

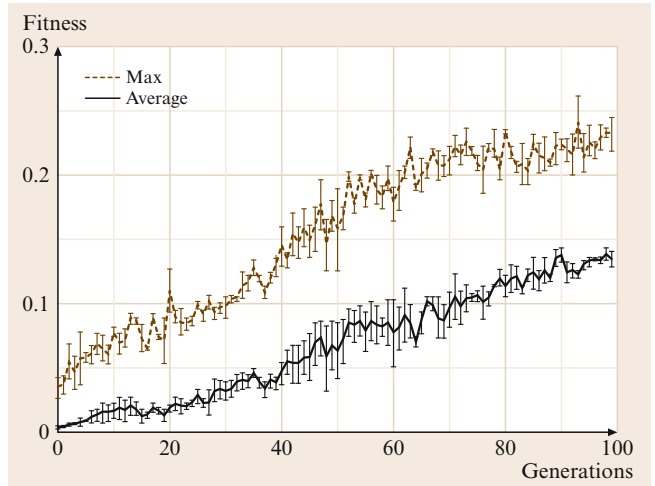


Fig. 61.3 Average fitness of the population and fitness of the best individual at each generation (error bars show standard error over three runs from different initial populations)

ary experiment was continued up to 200 generations. Further analysis revealed that this self-limitation of the navigation speed had an adaptive function because, considering the sensory and motor refresh rate together with the response profile of the distance sensors, robots that traveled faster had a higher risk of colliding with walls before detecting them; they gradually disappeared from the population.

Despite its simplicity, this experiments shows that evolution can discover solutions that match not only

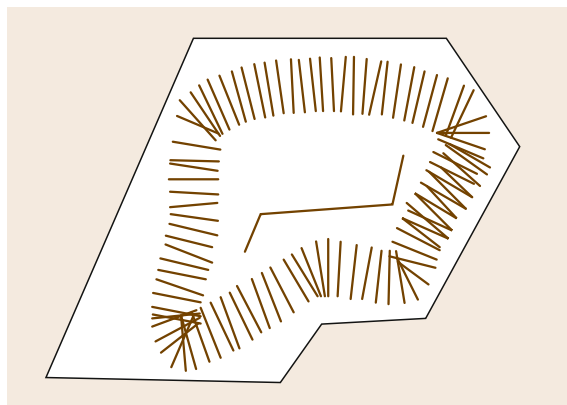


Fig. 61.4 Trajectory of the robot with the best neural controller of the last generation. Segments represent the axis between the two wheels. Data were recorded and plotted every 300 ms using an external laser positioning device

the computational requirements of the task to be solved, but also the morphological and mechanical properties of the robot in relation to its physical environment.

61.2.1 Evolution of Neural Controllers for Walking

Over the past 15 years or so, there has been a growing body of work on evolving controllers for various kinds of walking robots – a nontrivial sensorimotor coordination task. Early work in this area concentrated on evolving dynamical network controllers for simple (abstract) simulated insects (often inspired by cockroach studies) which were required to walk in simple environments [61.18, 19]. Earlier, *Beer* had introduced a neural architecture for locomotion based on studies of cockroaches [61.17], which is shown in Fig. 61.5. The promise of this work soon led to versions of this methodology being used on real robots. Probably the first success in this direction was by *Lewis et al.* [61.14, 20] who evolved a neural controller for a simple hexapod robot using coupled oscillators built from continuous-time, leaky-integrator, artificial neurons. All evaluations were done on the actual robot with each leg connected to its own pair of coupled neurons, leg swing being driven by one neuron and leg elevation by the other. These pairs of neurons were cross connected, in a manner similar to that used by *Beer and Gallagher* [61.19] (Fig. 61.5), to allow coordination between the legs. In order to speed up the process, they employed staged evolution where first an oscillator capable of moving a leg was evolved and then an architecture based on these oscillators was further evolved to develop walking. The robot was able to execute an efficient tripod gait on flat surfaces.

Gallagher et al. [61.21] described experiments where neural networks controlling locomotion in an artificial insect were evolved in simulation and then successfully downloaded onto a real hexapod robot. This machine was more complex than *Lewis et al.*'s, with a greater number of degrees of freedom per leg. In this approach, each leg was controlled by a fully connected network of five continuous-time, leaky-integrator neurons, each receiving a weighted sensory input from that leg's angle sensor. Initially the architecture shown in Fig. 61.5 was used, with the connection weights and neuron time constants and biases under genetic control. This produced efficient tripod gaits for walking on flat surfaces. In order to produce a wider range of gaits operating at a number of

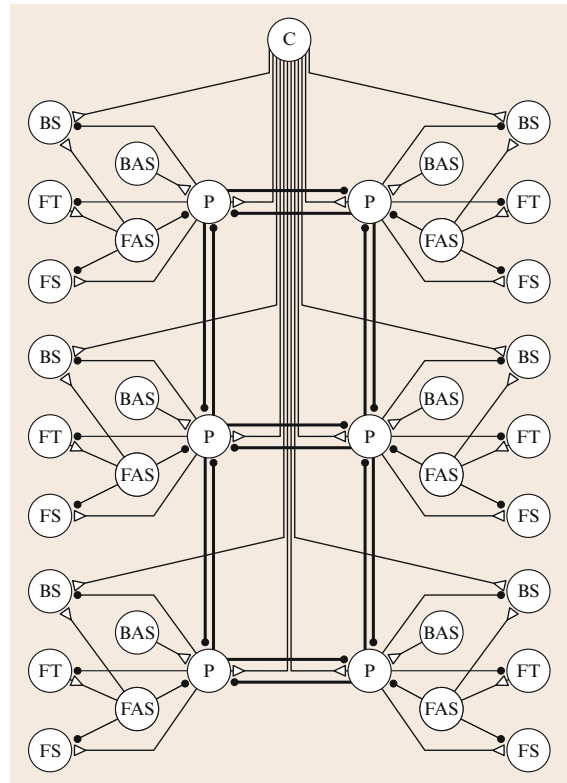


Fig. 61.5 Schematic diagram of a distributed neural network for the control of locomotion as used by *Beer et al.* [61.17]. Excitatory connections are denoted by *open triangles* and inhibitory connections are denoted by *filled circles*. C, command neuron; P, pacemaker neuron; FT, foot motor neuron; FS and BS, forward swing and backward swing motor neurons; FAS and BAS, forward and backward angle sensors

speeds such that rougher terrain could be successfully negotiated, a different distributed architecture, more inspired by stick insect studies, was found to be more effective [61.22].

Galt et al. [61.23] used a genetic algorithm to derive the optimal gait parameters for a Robug III robot, an eight-legged, pneumatically powered walking and climbing robot. The individual genotypes represented parameters defining each leg's support period and the timing relationships between leg movements. These parameters were used as inputs to a mechanistic finite-state machine pattern-generating algorithm that drove the locomotion. Such algorithms, which are often used in conventional walking machines, rely on relatively simple control dynamics and do not have the

same potential for the kind of sophisticated multigait coordination that complex dynamical neural network architectures, such as those described in this section, have been shown to produce. However, controllers were successfully evolved for a wide range of environments and to cope with damage and systems failure (although an individual controller had to be tuned to each environment; they were not able to self-adapt across a wide range of conditions). *Gomi and Ide* [61.24] evolved the gaits of an eight-legged robot (Fig. 61.6) using genotypes made of eight similarly organized sets of genes, each gene coding for leg motion characteristics such as the amount of delay after which the leg begins to move, the direction of the leg's motion, the end positions of both vertical and horizontal swings of the leg, and the vertical and horizontal angular speed of the leg. After a few dozen generations, where evaluation was on the robot, a mixture of tetrapod and wave gaits was obtained. Using the cellular encoding [61.25] developmental approach – which genetically encodes a grammar-tree program that controls the division of cells growing into a dynamical recurrent neural network of the kind used by Beer and colleagues – *Gruau* [61.26] evolved a single-leg neural controller for the same eight-legged robot used by Gomi and Ide. This generated a smooth and fast quadrupod locomotion gait. *Kodjabachian and Meyer* [61.27] extended this work to develop more sophisticated locomotion behaviors. *Jakobi* [61.28] successfully used his minimal simulation techniques (described in Sect. 61.3) to evolve controllers for the same eight-legged robot as

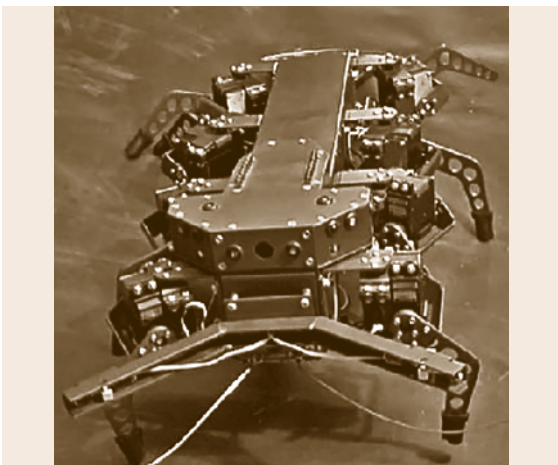


Fig. 61.6 The octopod robot built by Applied AI Systems Inc.

Gruau. Evolution in simulation took less than 2 h on what would today be regarded as a very slow computer, and was then successfully transferred to the real robot. *Jakobi* evolved modular controllers based on Beer's continuous recurrent networks to control the robot as it engaged in walking about its environment, avoiding obstacles and seeking out goals depending on the sensory input. The robot could smoothly change gait, move backward and forward, and even turn on the spot. More recent work has used similar architectures to those explored by the researchers mentioned above, to control more mechanically sophisticated robots such as the Sony Aibo [61.29].

Recently there has been successful work on evolving coupled oscillator style neural controllers for the highly unstable dynamic problem of biped walking. *Reil and Husbands* [61.30] showed that accurate physics based simulations using physics-engine software could be used to develop controllers able to generate successful bipedal gaits. Reil and colleagues have now significantly developed this technology to exploits its commercial possibilities, in the animation and games industries, for the real-time control of physically simulated three-dimensional (3-D) humanoid characters engaged in a variety of motor behaviors (see [61.31] for further details). Coupled neural oscillators have been evolved also to control the swimming pattern of articulated, snake-like, underwater robots using physics-based simulations [61.32].

Vaughan has taken related work in another direction. He has successfully applied evolutionary robotics techniques to evolve a simulation of a 3-D ten-degree-of-freedom bipedal robot. This machine demonstrates many of the properties of human locomotion. By using passive dynamics and compliant tendons, it conserves energy while walking on a flat surface. Its speed and gait can be dynamically adjusted and it is capable of adapting to discrepancies in both its environment and its body's construction [61.33]. The parameters of the body and continuous dynamical neural network controller were under genetic control. The machine started out as a passive dynamic walker [61.34] on a slope, and then throughout the evolutionary process the slope was gradually lowered to a flat surface. The machine demonstrated resistance to disturbance while retaining passive dynamic features such as a passive swing leg. This machine did not have a torso, but Vaughan has also successfully applied the method to a simplified two-dimensional (2-D) machine with a torso above the hips. When pushed, this dynamically stable bipedal machine

walks either forward or backwards just enough to release the pressure placed on it. It is also able to adapt to external and internal perturbations as well as variations in body size and mass [61.35].

McHale and *Husbands* [61.36, 37] have compared many forms of evolved neural controllers for bipedal and quadrupedal walking machines. Recurrent dynamical continuous time networks and GasNets (described in Sect. 61.6.3) were shown to have advantages in most circumstances. The vast majority of the studies

mentioned above were conducted for relatively benign environments. Notwithstanding this observation, we can conclude that the more complex dynamical neural network architectures, with their intricate dynamics, generally produce a wider range of gaits and generate smoother, more adaptive locomotion than the more standard use of systems based on finite-state machines employing parameterized rules governing the timing and coordination of individual leg movements [61.38].

61.3 Simulation and Reality

Few of the experiments in the previous section were carried out entirely on physical robots because

1. evolution may take a long time, especially if it is carried out on a single robot that incarnates the bodies of all the individuals of the evolving population;
2. the physical robot can be damaged because populations always contain a certain number of poorly performing individuals (for example, colliding against walls) by effect of random mutations;
3. restoring the environment to initial conditions between trails of different individuals or populations (for example, replenishing the arena with objects) may not always be feasible without human intervention;
4. evolution of morphologies and evolution of robots that can grow during their lifetime is almost impossible with today's technology without some level of human intervention.

For those reasons, researchers often resort to evolution in simulation and transfer the evolved controllers to the physical robot. In the case of morphology evolution, the physical robot is manually assembled according to the evolved specifications. However, it is well known that programs that work well in simulations may not function properly in the real world because of differences in sensing, actuation, and in the dynamic interactions between robot and environment [61.39]. This *reality gap* is even more evident in adaptive approaches, such as evolutionary robotics, where the control system and morphology are gradually crafted through the repeated interactions between the robot and the environment. Therefore, robots will evolve to match the specificities of the simulation, which differ from the real world. Al-

though these issues clearly rule out any simulation based on grid worlds or pure kinematics, over the last 10 years simulation techniques have dramatically improved and resulted in software libraries that model reasonably well dynamical properties such as friction, collision, mass, gravity, and inertia [61.40]. These software tools allow one to simulate articulated robots of variable morphology and their environment as fast as, or faster than, real time in a desktop computer. Today, those physics-based simulations are widely used by most researchers in evolutionary robotics and indeed most of the work with highly articulated robots is carried out with those simulations.

Nonetheless, even physics-based simulations include small discrepancies that can accumulate over time and result in very different behavior from reality (for example, a robot may get stuck against a wall in simulation whereas it can get free in reality, or vice versa). Also, physics-based simulations cannot account for diversity of response profiles of the individual sensors, motors, and gears of a physical robot. There are at least four methods to cope with these problems and improve the quality of the transfer from simulation to reality.

A widely used method consists of adding independent noise to the values of the sensors provided by the model and to the end position of the robot computed by the simulator [61.41]. Some software libraries allow the introduction of noise at several levels of the simulation. This solution prevents evolution from finding solutions that rely on the specificities of the simulation model. One may also sample the actual sensor values of the real robot positioned at several angles and distances from objects of different texture. Those values are then stored in a look-up table and retrieved with the addition of noise according to the position of the

robot in the environment [61.42]. This method proved to be very effective for generating controllers that transfer smoothly from simulation to reality. A drawback of this sampling method is that it does not scale up well to high-dimensional sensors (e.g., vision) or geometrically complicated objects.

Another method, also known as minimal simulations, consists of modeling only those characteristics of the robot and environment that are relevant for the emergence of desired behaviors [61.43]. These characteristics, which are referred to as base-set features, should be accurately modeled in simulation. Instead, all the other characteristics, which are referred to as implementation aspects, should be randomly varied across several trials of the same individual in order to ensure that evolving individuals do not rely on implementation aspects, but rely on base-set features only. Base-set features must also be varied to some extent across trials in order to ensure some degree of robustness of the individual with respect to base-set features, but this variation should not be so large that reliably fit controllers fail to evolve at all. This method allows very fast evolution of complex robot–environment situations, as in the example of the hexapod walk described in Sect. 61.2.1. A drawback of minimal simulations is that it is not always easy to tell in advance which are the base-set features that are relevant for the desired behavior.

Yet another method consists of the coevolution of the robot (control and/or morphology) and of the simulator parameters that are most likely to differ from the real world and that may affect the quality of the transfer [61.44]. This method consists of coevolving two populations, one encoding the properties of the robot and one encoding the parameters of the simulator. Coevolution happens in several passes through a two-stage

process. In stage one, a randomly generated population of robots are evolved in the default simulator and the best individual is tested on the real robot while the time series of sensory values are recorded. In stage two, the population of simulators is evolved to reduce the difference between the time series recorded on the real robot and the time series obtained by testing evolved robots within the simulator. The best evolved simulator is then used for stage one where a new randomly generated population is evolved and the best individual is tested on the real robot to generate the time series for stage two of simulator evolution. This two-stage coevolution is repeated several times until the error between simulated and real robot behavior is the smallest possible. It has been shown that approximately 20 passes of the two-stage process are sufficient to evolve a good control system that could be transferred to an articulated robot. In that case, the real robot was used to test only 20 individuals.

Finally, another method consists of genetically encoding and evolving the learning rules of the control system, rather than its parameters (e.g., connection strengths). The parameters of the decoded control system are always initialized to small random values at the beginning of an individual lifetime and must self-organize using the learning rules [61.45]. This method prevents evolution from finding a set of control parameters that fit the specificities of the simulation model, and encourages emergence of control systems that remain adaptive to partially unknown environments. When such an evolved individual is transferred to the real robot, it will develop online its control parameters according to the genetically evolved learning rules and taking into account the specificities of the physical world. This method is described in more detail in Sect. 61.7.2 on evolution of learning.

61.4 Simple Controllers, Complex Behaviors

Behavior is a dynamical process resulting from nonlinear interactions (occurring at a fast time rate) between the agent's control system, its body, and the environment [61.46, 47]. At any time step, the environment and the agent–environment relation influence the body and the motor reaction of the agent, which in turn influences the environment and/or the agent–environmental relation (Fig. 61.7). Sequences of these interactions lead to a dynamical process where the contributions of the different aspects (i.e., the robot's control system, the

robot's body, and the environment) cannot be separated. This implies that even complete knowledge of the elements governing the interactions provides little insight into the behavior emerging from these interactions [61.48, 49].

An important advantage of evolutionary robotics is that it is not necessary to identify the relations between the rules governing the interactions and the resulting behavior [61.1, 49]. Evolutionary robotics is an adaptation process where the free parameters of the robots that

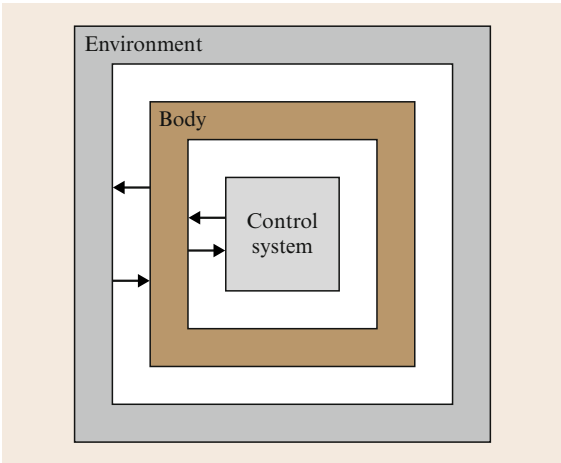


Fig. 61.7 Robot behavior results from nonlinear interactions, occurring at fast time rates, between the agent’s control system, its body, and the environment

regulate the interactions, initially randomly assigned, are modified through a process of random variation and are selected and/or discarded on the basis of their effects at the behavioral level. These characteristics allow evolving robots to discover useful behavioral properties emerging from the interactions without the need to identify the relations between the rules governing the interaction and the resulting behavior. An emergent behavioral property or behavior is a form of behavior that can hardly be predicted or inferred by an external observer even when they have complete knowledge of the interacting elements and of the rules governing those interactions. The possibility of developing robots that exploit emergent behavior, in turn, allows evolutionary methods to come up with simple solutions to problems that are complex from an observer’s perspective.

As an example (Fig. 61.8), consider the case of a Khepera robot placed in an arena surrounded by walls and containing a food object (i.e., a cylindrical object) that the robot should find and remain close to [61.50]. The robot is provided with eight infrared sensors and two motors controlling the two corresponding wheels. From the point of view of an external observer, solving this problem requires robots able to:

1. explore the environment until an obstacle is detected
2. discriminate whether the obstacle detected is a wall or a cylindrical object

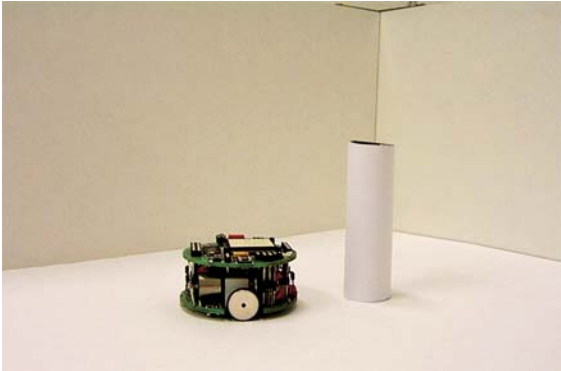


Fig. 61.8 The environment and the robot. The environment consists of an arena of 60 × 35 cm and contains a cylindrical objects placed at a randomly selected location

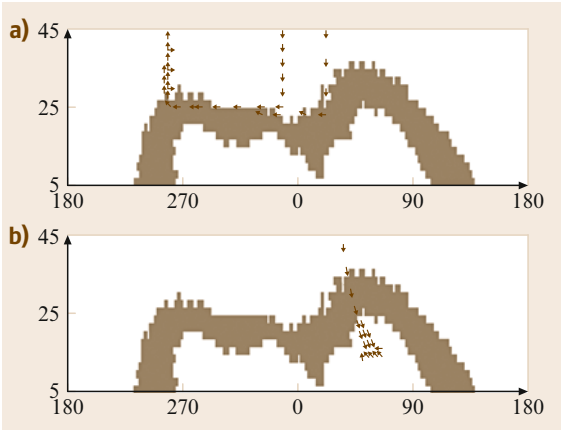


Fig. 61.9a,b Angular trajectories of an evolved robot close to a wall (a) and to a cylinder (b). The picture was obtained by placing the robot at a random position in the environment, leaving it free to move for 500 cycles, and recording its relative movements with respect to the two types of objects for distances smaller than 45 mm. For sake of clarity, *arrows* are used to indicate the relative direction, but not the amplitude of movements

3. approach or avoid the object depending on the object type

A detailed analysis of the sensory patterns experienced by the robot indicated that the task of discriminating the two objects is far from trivial since the two classes of sensory patterns experienced by

robots close to a wall and close to cylindrical objects overlap significantly. However, robots evolved for the ability to solve this task resorted to a strategy that does not require explicit discrimination of the two types of objects [61.50].

In all replications of the experiment, the evolved robot moved away from walls, but when they encountered the food object tended to oscillate back and forth or left and right in its proximity (Fig. 61.9). This solution consists of producing a behavioral attractor near the cylindrical object. A behavioral attractor is a series of sensorimotor sequences that maintain the robot close to the object. In this case, the forward movement in front of the cylindrical object generates a variation of the sensory

pattern experienced by the robot that, in turn, triggers a backward movement.

Therefore, evolved robots do not solve the problem by discriminating the two type of objects (cylinder and wall) and displaying an approaching or avoiding behavior, but rather exploit behaviors that emerge from the interaction between the robot's control system, the robots' body, and the environment. The possibility to discover and rely on these forms of emergent behavior allows evolving robots to find computationally simple solutions to apparently complex problems. Indeed, the problem described in this section only requires a simple reactive neural controller with one layer of feedforward connections between sensors and motors.

61.5 Seeing the Light

The experiments described so far rely mainly on relatively simple distance sensors, such as active infrared or sonar. Pioneering experiments on evolving visually guided behaviors were performed at Sussex University [61.51] on a specially designed gantry robot (Fig. 61.10). Discrete-time dynamical recurrent neural networks and visual sampling morphologies were concurrently evolved: the brain was developed in tandem with the visual sensor [61.13, 52, 53]. The robot was designed to allow real-world evolution by having *off-board* power and processing so that the robot could be run for long periods while being monitored by automatic

fitness evaluation functions. A charge-coupled device (CCD) camera points down towards a mirror angled at 45° as shown in Fig. 61.10. The mirror can rotate around an axis perpendicular to the camera's image plane. The camera is suspended from the gantry, allowing motion in the X , Y , and Z dimensions. This effectively provides an equivalent to a wheeled robot with a forward-facing

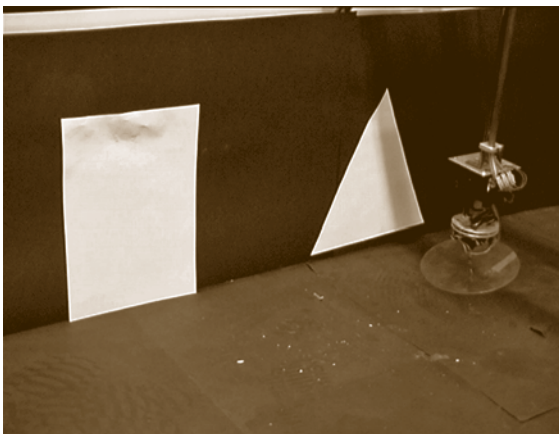


Fig. 61.10 The gantry robot used in the visual discrimination task. The camera inside the top box points down at the inclined mirror, which can be turned by the stepper motor beneath. The lower plastic disk is suspended from a joystick to detect collisions with obstacles

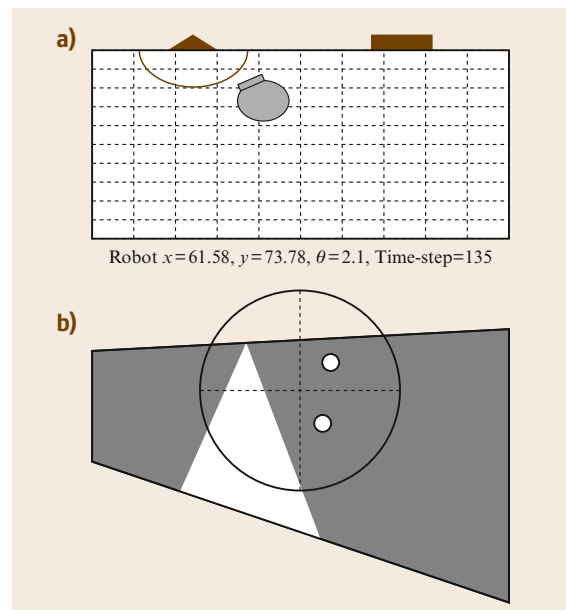


Fig. 61.11a,b The shape discrimination task. **(a)** The position of the robot in the arena, showing the target area in front of the triangle. **(b)** The robot camera's field of view showing the visual patches selected by evolution for sensory input

camera when only the X and Y dimensions of translation are used. The additional dimension allows flying behaviors to be studied.

The apparatus was initially used in a manner similar to the real-world experiments on navigation in the looping maze with the miniature mobile robot described in Sect. 61.2. A population of strings encoding robot controllers and visual sensing morphologies were stored on a computer to be downloaded one at a time onto the robot. The exact position and orientation of the camera head can be accurately tracked and used in the fitness evaluations. A number of visually guided navigation behaviors were successfully achieved, including navigating around obstacles, tracking moving targets, and discriminating between different objects [61.52]. The evolutionary process was incremental. The ability to distinguish between two different targets was evolved on top of the single target-finding behavior. The chromosome was of dynamic length so the neurocontroller was structurally further developed by evolution to achieve the new task (neurons and connections added). In the

experiment illustrated in Figs. 61.10 and 61.11, starting from a random position and orientation, the robot had to move to the triangle rather than the rectangle. This had to be achieved irrespective of the relative positions of the shapes and under very noisy lighting conditions. Recurrent neural network controllers were evolved in conjunction with visual sampling morphologies. Only genetically specified patches from the camera image were used (by being connected to input neurons according to the genetic specification). The rest of the image was thrown away. This resulted in extremely minimal systems using only two or three pixels of visual information, yet still able to perform the task reliably under highly variable lighting conditions [61.13, 52].

This was another example of staged, or incremental, evolution to obtain control systems capable of solving problems that are either too complex or may profit from an evolutionary methodology that discovers, preserves, and builds upon subcomponents of the solution. For an evolutionary method that incorporate strategies to explicitly address this issue, interested readers may refer to [61.54]. However, staged evolution remains a poorly explored area of evolutionary robotics that deserves further study and a more principled approach [61.55] in order to achieve increasingly complex robotic systems.

61.5.1 Coevolution of Active Vision and Feature Selection

Machine vision today can hardly compete with biological vision despite the enormous power of computers. One of the most remarkable – and often neglected – differences between machine vision and biological vision is that computers are often asked to process an entire image in one shot and produce an immediate answer whereas animals take time to explore the image over time, searching for features and dynamically integrating information over time.

Active vision is the sequential and interactive process of selecting and analyzing parts of a visual scene [61.56–58]. *Feature selection* instead is the development of sensitivity to relevant features in the visual scene to which the system selectively responds, e.g., [61.59]. Each of these processes has been investigated and adopted in machine vision. However, the combination of active vision and feature selection is still largely unexplored. An intriguing hypothesis is that coevolution of active vision and feature selection could greatly simplify the computational complexity of vision-based behavior by facilitating each other's task.

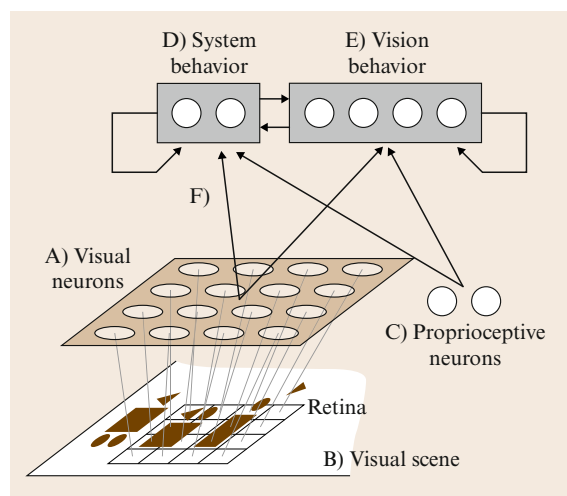


Fig. 61.12 The neural architecture of the active vision system is composed of: (A) a grid of visual neurons with nonoverlapping receptive fields whose activation is given by (B) the grey level of the corresponding pixels in the image; (C) a set of proprioceptive neurons that provide information about the movement of the vision system; (D) a set of output neurons that determine the behavior of the system (pattern recognition, car driving, robot navigation); (E) a set of output neurons that determine the behavior of the vision system; and (F) a set of evolvable synaptic connections. The number of neurons in each subsystem can vary according to the experimental settings

This hypothesis was investigated in a series of experiments [61.60] on coevolution of active vision and feature selection for behavioral systems equipped with a primitive moving retina and a deliberately simple neural architecture (Fig. 61.12). The neural architecture was composed of an artificial retina and two sets of output units. One set of output units determined the movement and zooming factor of the retina, and the other set of units determined the behavior of the system, such as the response of a pattern-recognition system, the control parameters of a robot, or the actions of a car driver. The neural network was embedded in a behavioral system and its input/output values were updated every 300 ms while its fitness was computed. Therefore, the synaptic weights of this network were responsible for both the visual features on which the system based its behavior and for the motor actions necessary to search for those features.

In a first set of experiments, the neural network was embedded in a simulated pan-tilt camera and asked to discriminate between triangles and squares of different size that could appear at any location of a screen (Fig. 61.13a), a perceptual task similar to that explored with the gantry robot described in Sect. 61.5. The visual system was free to explore the image for 60 s while continuously reporting whether the current screen showed a triangle or a square. The fitness was proportional to the amount of correct responses accumulated over the 60 s for several screenshots containing various instances of the two shapes. Evolved systems were capable of correctly identifying the type of shape with 100% accuracy after a few seconds despite the fact that this recognition problem is not linearly separable and that the neural network does not have hidden units, which in theory are necessary to solve nonlinearly separable tasks. Indeed, the same neural network presented with the same set of images and trained with supervised learning, but without the possibility to actively explore the scene, was not capable of solving the task. The evolved active vision system developed sensitivity to vertical edges, oriented edges and corners, and used its movement to search for these features in order to tell whether the shape was a triangle or a square. These features, which are also found in the early visual system of almost all animals, are invariant to size and location.

In a second set of experiments, the neural network was embedded in a simulated car and was asked to drive over several mountain circuits (Fig. 61.13b). The simulator was a modified version of a car race video game. The neural network could move the retina across the scene seen through the windscreen at the driver's seat

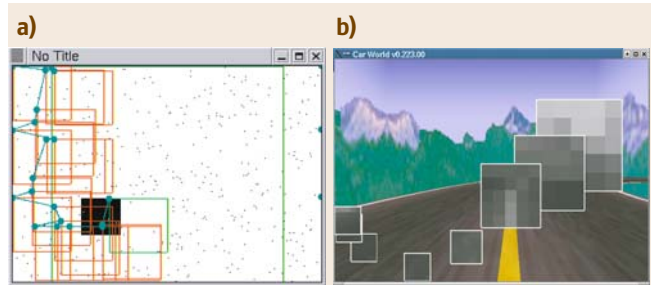


Fig. 61.13 (a) An evolved individual explores the screen searching for the shape and recognizes it by the presence of a vertical edge. (b) Search for the edge of the road at the beginning of a drive over a mountain road



Fig. 61.14 A mobile robot with a pan-tilt camera is asked to move within the walled arena in the office environment

and control the steering, acceleration, and braking of the car. The fitness was inversely proportional to the time taken to complete the circuits without exiting the road. Evolved networks completed all circuits with time laps competitive to those of well-trained students controlling the car with a joystick. The evolved network started by searching for the edge of the road and tracked its relative position with respect to the edge of the windscreen in order to control steering and acceleration. This behavior was supported by the development of sensitivity to oriented edges.

In a third set of experiments, the neural network was embedded in a real mobile robot with a pan-tilt camera that was asked to navigate in a square arena with low walls located in an office (Fig. 61.14). The fitness was proportional to the amount of straight motion measured over two minutes. Robots that hit the walls because they watched people or other irrelevant features of the office had lower fitness than robots that could perform long

straight paths and avoid walls of the arena. Evolved robots tended to fixate the edge between the floor and the walls of the arena, and turned away from the wall when the size of its retinal projection became larger than a threshold. This combination of sensitivity to oriented edges and looming is also found in the visual circuits of several insects and birds.

In a further set of experiments [61.61], the visual pathway of the neural network was augmented by an intermediate set of neurons whose synaptic weights could be modified by Hebbian learning [61.62] while the robot moved in the environment. All the other synaptic weights were genetically encoded and evolved. The results showed that lifelong development of the recep-

tive fields improved the performance of evolved robots and allowed robust transfer of evolved neural controllers from simulated to real robots, because the receptive fields developed sensitivity to features encountered in the environment where they happen to be born (see also the section above on simulation and reality). Furthermore, the results showed that the development of visual receptive fields was significantly and consistently affected by active vision as compared to the development of receptive fields passively exposed to the same set of sample images. In other words, robots evolved with active vision developed sensitivity to a smaller subset of features in the environment and actively tracked those features to maintain a stable behavior.

61.6 Computational Neuroethology

Evolutionary robotics is also used to investigate open questions in neuroscience and cognitive science [61.65] because it offers the vantage point of a behavioral system that interacts with its environment [61.66]. Although the results should be carefully considered when drawing analogies with biological organisms, evolutionary

robotics can generate and test hypotheses that could be further investigated with mainstream neuroscience methods.

For example, the active vision system with Hebbian plasticity described in the previous section was used to answer a question raised by *Held* and *Hein* [61.63] in the 1960s. The authors devised the apparatus shown in Fig. 61.15 where the free movements of a kitten (*active kitten*) were transmitted to a second kitten that was carried in a gondola (*passive kitten*). The second kitten could move its head, but its feet did not touch the ground. Consequently, the two kitten received almost identical visual stimulation, but only one of them received that stimulation as a result of body self-movement. After a few days in that environment, only the active kitten displayed normal behavior in several visually guided tasks. The authors suggested the hypothesis that proprioceptive motor information resulting from generation of actions was necessary for the development of normal, visually guided behavior.

The kitten experiments were replicated by cloning an evolved robot controller and randomly initializing the synaptic values of the adaptive visual pathways in both clones. One cloned robot was then left free to move in a square environment while the other cloned robot was forced to move along imposed trajectories, but was free to control its camera position, just like the passive kitten [61.67]. The results indicated that the visual receptive fields and behaviors of passive robots differ significantly from those of active robots. Furthermore, passive robots that were later left free to move were no longer capable of properly avoiding walls. A thorough analysis of neural activation correlated with behavior of the robot

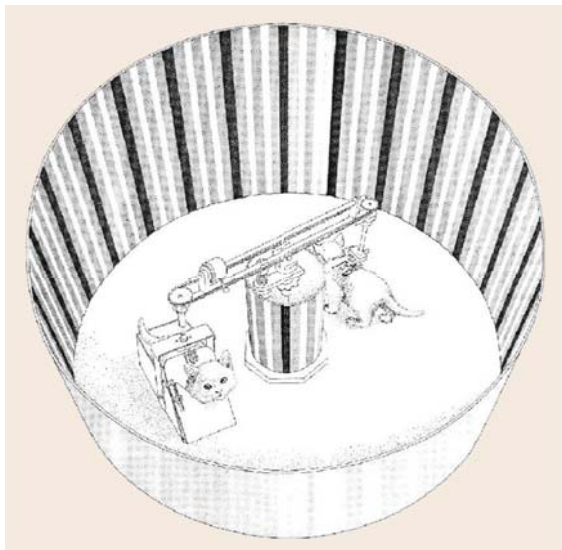


Fig. 61.15 The original apparatus in [61.63], where the gross movements of a kitten moving almost freely were transmitted to a second kitten that was carried in a gondola. Both kittens were allowed to move their head. They received essentially the same visual stimulation because of the unvarying pattern on the walls and the center post of the apparatus (after [61.64], with permission)

and even transplantation of neurons across active and passive robots revealed that the poor performance was due to the fact that passive robots could not completely select the visual features they were exposed to. Consequently, passive robots developed sensitivity to features that were not functional to their normal behavior and interfered with other dominant features in the visual field. Whether this explanation also holds for living animals remains to be further investigated, but at least these experiments indicated that motor feedback is not necessary to explain the pattern of pathological behavior observed in animals and robots.

61.6.1 Emergence of Place Cells

Let us now consider the case of an animal exploring an environment and periodically returning to its nest to feed. It has been speculated that this type of situation requires the formation of spatial representations of the environment that allow the animal to find its way home [61.68]. Different neural models with various degrees of complexity and biological detail that could provide such functionality have been proposed [61.69, 70].

Would a robot evolved under similar survival conditions develop a spatial representation of the environment and, if so, what type of representation would that be? These questions were explored using the same Khepera robot and evolutionary methodology described in Sect. 61.2 for reactive navigation in the looping maze. The environment was a square arena with a small patch on the floor in a corner where the robot could instantaneously recharge its (simulated) battery (Fig. 61.16). The environment was located in a dark room with a small light tower over the *recharging station*.

The sensory system of the robot was composed of eight distance sensors, two ambient-light sensors (one on each side), one floor-color sensor, and a sensor for battery charge level. The battery lasted only 20 s and had a linear discharge. The evolutionary neural network included five fully connected internal neurons between sensory and motor neurons. The same fitness function described in Sect. 61.2 for navigation in the looping maze was used, except for the middle term which had been used to encourage straight navigation in the looping maze. The fitness value was computed every 300 ms and accumulated over the life span of the individual. Therefore, individuals who discovered where the charger was could live longer and accumulate more fitness by exploring the environment (individuals were killed if they survived longer than 60 s to limit the experimentation time).

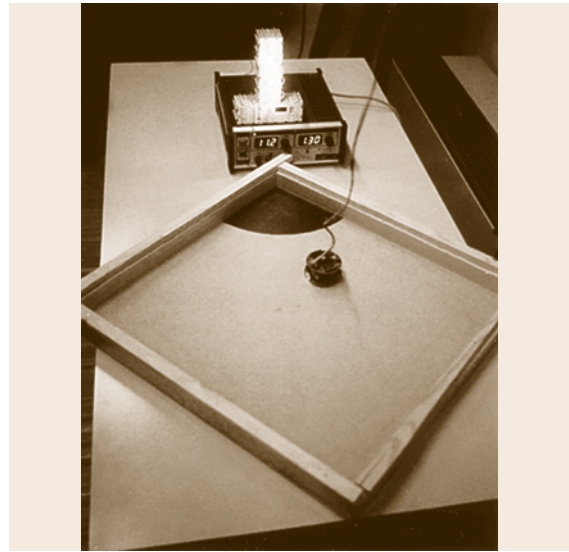


Fig. 61.16 Bird's eye view of the arena with the light tower over the recharging station and the Khepera robot

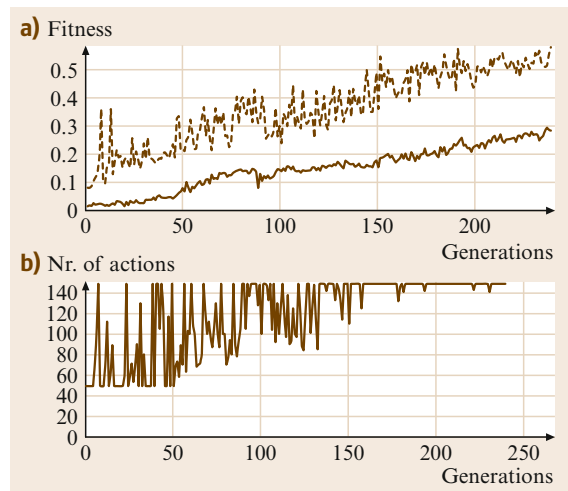


Fig. 61.17 (a) Average population fitness (*continuous line*) and fitness of the best individual (*dotted line*). (b) Life span of the best individuals measured as number of sensorimotor cycles, or actions. Individuals start with a full battery which lasts 50 actions (20 s), if not recharged. The maximum life span is 150 actions

The same physical robot evolved for 10 days and nights as both the fitness and life span of individuals continued to increase (Fig. 61.17). After approximately 200 generations, the robot was capable of navigating around the environment, covering long trajectories while avoid-

ing both walls and the recharging area. When the battery was almost discharged it initiated a straight navigation towards the recharging area and exited immediately after battery recharge to resume navigation. Best evolved individuals always entered the recharging area one or two seconds before full discharge of the battery. That implies that robots must somehow calibrate the timing and trajectory of their homing behavior depending on where they happened to be in the environment.

In order to understand how that behavior could possibly be generated, a set of neuroethological measures were performed using a laser positioning device that provided exact position and orientation of the robot every 300 ms. By correlating the robot position and behavior with the activation of the internal neurons in real time while the evolved individual freely moved in the environment, it was possible to see that some neurons specialized for reactive behaviors, such as obstacle avoidance, forward motion, and battery monitoring. Other neurons instead displayed more complex activation patterns. One of them revealed a pattern of activation levels that depended on whether the robot was oriented facing the light tower or facing the opposite direction (Fig. 61.18). In the former case, the activation pattern

reflected zones of the environment and paths typically followed by the robot during exploration and homing. For example, the robot trajectory towards the recharging area never crossed the two *gate walls* visible in the activation maps around the recharging station. When the robot faced the opposite direction, the same neuron displayed a gradient field orthogonally aligned with the recharging area. This gradient provides an indication of the distance from the recharging area. Interestingly, this pattern of activity is not significantly affected by the charge level of the battery.

The functioning of this neuron reminds of the classic findings on the hippocampus of the rat brain where some neurons (also known as *place cells*) selectively fire when the rat is in specific areas of the environment [61.71]. Also, the orientation-specific pattern of neural activation measured on the evolved robot is reminiscent of the so-called *head-direction neurons* in the rat hippocampus, which are positioned nearby place cells, whose firing patterns depend on the rat heading direction with respect to an environmental landmark [61.72]. Although the analogy between brains of evolved robots and of biological organisms should not be taken too literally, these results indicate that the two organisms converge towards a functionally similar neural strategy, which may be more efficient to address this type of situation than a strategy that does not rely on representations (but only on reactive strategies such as random motion, light following, or dead reckoning).

61.6.2 Spiking Neurons

The great majority of biological neurons communicate using self-propagating electrical pulses called *spikes*, but from an information-theoretic perspective it is not yet clear how information is encoded in the spike train. Connectionist models [61.73], by far the most widespread, assume that what matters is the *firing rate* of a neuron, that is, the average quantity of spikes emitted by the neuron within a relatively long time window (for example, over 100 ms). Alternatively, what matters is the average number of spikes of a small population of neurons at a give point. In these models the real-valued output of an artificial neuron represents the firing rate, possibly normalized relatively to the maximum attainable value. Pulsed models [61.74], instead, are based on the assumption that the *firing time*, that is, the precise time of emission of a single spike, may convey important information [61.75]. Spiking neuron models have slightly more complicated dynamics of synaptic and membrane integration. Depending on one's theory

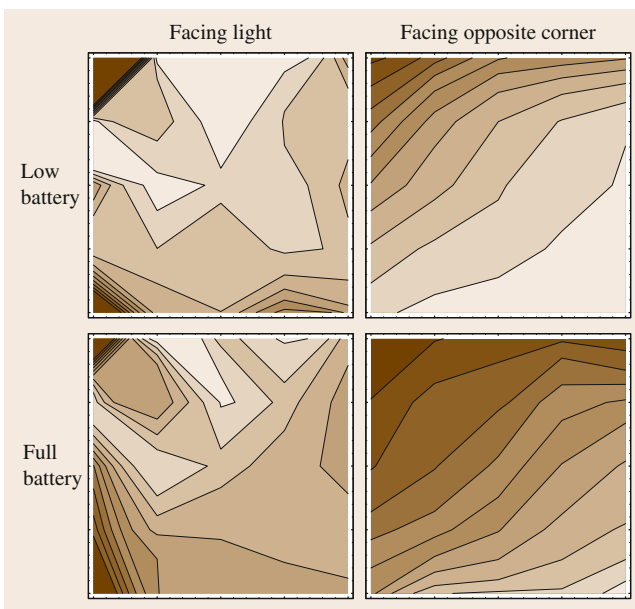


Fig. 61.18 Activation levels (brightness proportional to activation) of an internal neuron plotted over the environment while the robot was positioned at various locations in each of the four conditions (facing recharging area or not, discharged battery or not). The recharging area is located at the *top left corner* of each map

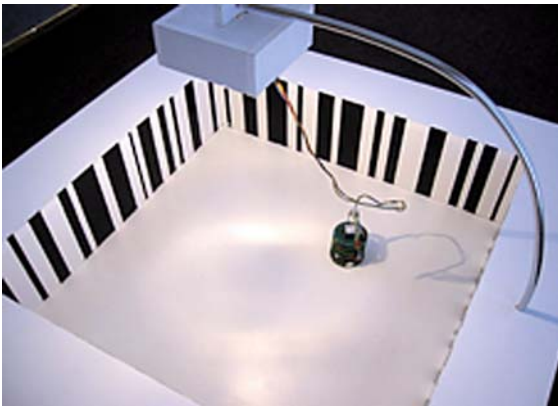


Fig. 61.19 A network of spiking neurons is evolved to drive the vision-based robot in the arena. The light below the rotating contacts allows continuous evolution also overnight

of what really matters, connectionist or spiking models are used.

However, designing circuits of spiking neurons that display a desired functionality is still a challenging task. The most successful results in the field of robotics obtained so far focused on the first stages of sensory processing and on relatively simple motor control [61.76, 77]. Despite these implementations, there are not yet methods for developing complex spiking circuits that could display minimally cognitive functions or learn behavioral abilities through autonomous interaction with a physical environment.

Artificial evolution represents a promising methodology to generate networks of spiking circuits with desired functionalities expressed as behavioral criteria (fitness function). Evolved networks could then be examined to detect what communication modality is used and how that correlates with observed behavior of the robot.

Floreano and colleagues [61.78] evolved a fully connected network of spiking neurons for driving a vision-based robot in an arena painted with black stripes of variable size against a white background (Fig. 61.19). The Khepera robot used in these experiments was equipped with a vision turret composed of one linear array of grayscale photoreceptors spanning a visual field of 36° . The output values of a bank of local contrast detection filters were converted in spikes (the stronger the contrast, the larger the number of spikes per second) sent to ten fully connected spiking neurons implemented according to the spike response model [61.79]. The spike series of a subset of these neurons was translated into motor commands (more spikes per second corresponded

to faster rotation of the wheel). The fitness function was the amount of forward translation of the robot measured over 2 min. Consequently robots that turned in place or hit the walls had comparatively lower fitness than robots that could move straight and turn only to avoid walls. The genome of these robots was a bit string that encoded only the sign of the neurons and the presence of synaptic connections. Existing connections were set to 1 and could not change during the lifetime of the robot.

Evolution reliably discovered very robust spiking controllers in approximately 20 generations (approximately 30 h of evolution on the real robot). Evolved robots could avoid not only the walls, but any object positioned in front of them. Detailed analysis of the best evolved controllers revealed that neurons did not exploit time differences between spikes, which one would have expected if optic flow was used to detect distance from walls. Instead, they simply used the number of incoming spikes (firing rate) as an indication of when to turn. When the robot perceived a lot of contrast it would go straight, but when the contrast decreased below a certain threshold (indicating that it approached an object), it started to turn away. This extremely efficient and simple result seems to be in contrast with theories of optic flow detection in insects and may be worth considering as an alternative hypothesis for vision-based behavior.

Spiking neural networks turned out to be more evolvable than connectionist models (at least for this task). One possible explanation is that spiking neurons have subthreshold dynamics that, to some extent, can be affected by mutations without immediately affecting the output of the network.

The robust results and compact genetic encoding encouraged the authors to use an even simpler model of spiking neuron so that the entire neural network could be mapped in less than 50 bytes of memory. The evolutionary algorithm was also reduced to a few lines of code and the entire system was implemented within

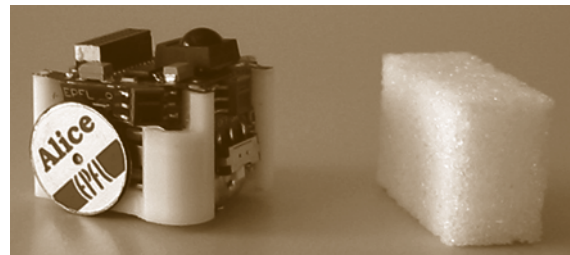


Fig. 61.20 The Alice sugar-cube robot equipped with the evolutionary spiking neural network implemented within its PIC microcontroller

a programmable intelligent computer (PIC) microcontroller without the need for any external computer for data storage. The system was used for a sugar-cube robot (Fig. 61.20) that autonomously and reliably developed the ability to navigate around a maze in less than an hour [61.80]. Interestingly, evolved spiking controllers developed a pattern of connections where spiking neurons received connections from a small patch of neighboring sensors, but not from other sensors, and were connected only to neighboring spiking neurons. This pattern of connectivity is also observed in biological systems and encourages specialization of neurons to sensory features.

61.6.3 GasNets

This section describes another style of artificial neural network strongly inspired by those parts of contemporary neuroscience that emphasize the complex electrochemical nature of real nervous systems. In particular, they make use of an analogue of *volume signaling*, whereby neurotransmitters freely diffuse into a relatively large volume around a nerve cell, potentially affecting many other neurons [61.81, 82]. This exotic form of neural signaling does not sit easily with classical pictures of brain mechanisms and is forcing a radical rethink of existing theory [61.83–86]. The class of artificial neural networks developed to explore artificial volume signaling are known as GasNets [61.87]. These are essentially standard neural networks augmented by a chemical signaling system comprising a diffusing *virtual gas* which can modulate the response of other neurons. A number of GasNet variants, inspired by different aspects of real nervous systems, have been explored in an evolutionary robotics context as artificial nervous systems for mobile autonomous robots. They have been shown to be significantly more evolvable, in terms of speed to a good solution, than other forms of neural networks for a variety of robot tasks and behaviors [61.36, 87–89]. They are being investigated as potentially useful engineering tools and as a way of gaining helpful insights into biological systems [61.85, 90, 91].

By analogy with biological neuronal networks, GasNets incorporate two distinct signaling mechanisms, one *electrical* and one *chemical*. The underlying *electrical* network is a discrete-time-step recurrent neural network with a variable number of nodes. These nodes are connected by either excitatory or inhibitory links.

In addition to this underlying network in which positive and negative *signals* flow between units, an abstract

process loosely analogous to the diffusion of gaseous modulators is at play. Some units can emit virtual *gases* which diffuse and are capable of modulating the behavior of other units by changing the profile of their output functions. The networks occupy a 2-D space; the diffusion processes mean that the relative positioning of nodes is crucial to the functioning of the network. Spatially, the gas concentration varies as an inverse exponential of the distance from the emitting node with a spread governed by a parameter r with the concentration set to zero for all distances greater than r . The total concentration of gas at a node is determined by summing the contributions from all other emitting nodes.

For mathematical convenience, in the original GasNet there are two *gases*, one whose modulatory effect is to increase the transfer function gain parameter and one whose effect is to decrease it. Thus the gas does not alter the electrical activity in the network directly but rather acts by continuously changing the mapping between input and output for individual nodes, either directly or by stimulating the production of further virtual gas. The general form of the diffusion is based on the properties of a (real) single-source neuron as modeled in detail by *Philippides* et al. [61.85, 90]. The modulation chosen is motivated by what is known of NO modulatory effects at synapses [61.92]. For full details see [61.87].

Various extensions of the basic GasNet have been produced. Two in particular are strongly inspired by contemporary neuroscience. The *plexus model* is directly inspired by a type of signaling seen in the mammalian cerebral cortex in which the NO signal is generated by the combined action of many fine NO-producing fibers, giving a targeted *cloud* which is distant from the neurons from which the fiber plexus emanates [61.91]. In the plexus GasNet, which models this form of signaling at an abstract level, the spatial distribution of gas concentration has been modified to be uniform over the area of affect. The center of this gas diffusion cloud is under genetic control and can be distant from the controlling node (which, by analogy, is the source of the *plexus*) [61.89]. All other details of the models are identical to the original GasNet model, as described earlier. The *receptor GasNet* incorporates an aspect of biological neuronal networks that has no analog in the vast majority of ANNs: the role of receptor molecules. Although neuroscience is a long way from a full understanding of receptor mechanisms, a number of powerful systems level ideas can be abstracted.

Details of the receptor variant are similar to the basic GasNet except there is now *only one* virtual gas and each node in the network can have one of three discrete quan-

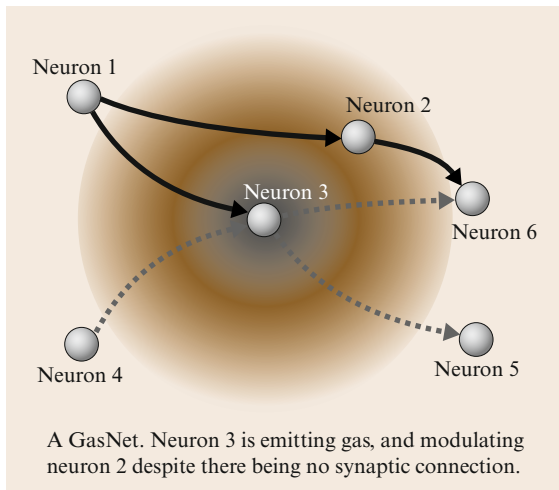


Fig. 61.21 A basic GasNet showing positive (*solid*) and negative (*dashed*) electrical connections and a diffusing virtual gas creating a *chemical* gradient

ties (zero, medium, maximum) of a number of possible receptors. The modulation the diffusing neurotransmitter affects at a neuron depends on which receptors are present. The strength of a modulation at a node is proportional to the product of the gas concentration at the node and the relevant receptor quantity. In the original GasNet, any node that was in the path of a diffusing transmitter would be modulated in a fixed way. The receptor model allows site-specific modulations, including no modulation (no receptors) and multiple modulations at a single site (see [61.89] for further details).

Although most of the GasNet variants described in this section have been successfully used in a number of robotic tasks, their evolvability and other properties were thoroughly compared on a version of the (gantry) robot visual discrimination task described in Sect. 61.5. All aspects of the networks were under genetic control: the number of nodes, the connectivity and, in the case of the GasNets, all parameters governing volume signaling (including the position of the nodes and whether or not they were *virtual gas* emitters). The visual sampling morphology was also under evolutionary control. The original basic GasNet was found to be significantly more

evolvable than a variety of other styles of connectionist neural networks as well as a GasNet with the volume signaling disabled. Successful GasNet controllers for this task tended to be rather minimal, in terms of numbers of nodes and connections, while possessing complex dynamics [61.87]. Later experiments comparing the basic GasNet with the plexus and receptor variants showed the latter two to be considerably more evolvable than the former, with the receptor GasNet being particularly successful [61.89].

The GasNet experiments mentioned above demonstrated that the intricate network dynamics made possible by the artificial volume signaling mechanisms can be readily harnessed to generate adaptive behaviors in autonomous agents. They also throw up such questions as why GasNets are more evolvable than many other forms of ANN and why there is a difference in evolvability between GasNet variants. Investigations of this question indicate that the interaction between the two GasNet signaling mechanisms, *electrical* and *chemical*, plays a crucial role [61.88, 89]. Evolutionary theory led to the hypothesis that systems involving distinct yet coupled processes are highly evolvable when the coupling is flexible (i.e., it is relatively easy for evolution to change the degree of coupling in the system) with a bias towards a loose coupling; this allows the possibility of *tuning* one process against the other without destructive interference [61.88, 89, 93]. This may also be the case for subthreshold dynamics of spiking neural networks, which although not yet compared to GasNets, were shown to be more evolvable than connectionist networks. Measurements of the degree of coupling in the GasNets variants versus speed of evolution supported this view [61.89]; the receptor GasNet, for which the evolutionary search process has the most direct control over the degree of coupling between the signaling processes, and which has a bias towards a loose coupling, was by far the most evolvable [61.89]. These and ongoing investigations indicate that explicitly dealing with the electrochemical nature of nervous systems is likely to be an increasingly fruitful area of research, both for evolutionary robotics and for neuroscience, that will likely force us to broaden our notions of what behavior-generating mechanisms might look like.

61.7 Evolution and Learning

Evolution and learning (or phylogenetic and ontogenetic adaptation) are two forms of biological adaptation that

differ in space and time. Evolution is a process of selective reproduction and substitution based on the existence

of a population of individuals displaying variability at the genetic level. Learning, instead, is a set of modifications taking place within each single individual during its own life time. Evolution and learning operate on different time scales. Evolution is a form of adaptation capable of capturing relatively slow environmental changes that might encompass several generations (e.g., the perceptual characteristics of food sources for a given species). Learning, instead, allows an individual to adapt to environmental modifications that are unpredictable at the generational level. Learning might include a variety of mechanisms that produce adaptive changes in an individual during its lifetime, such as physical development, neural maturation, variation of the connectivity between neurons, and synaptic plasticity. Finally, whereas evolution operates on the genotype, learning affects only the phenotype, and phenotypic modifications cannot directly modify the genotype.

Researchers have combined evolutionary techniques and learning techniques (supervised or unsupervised learning algorithm such as reinforcement learning or Hebbian learning; for a review see [61.94]). These studies have been conducted with two different purposes:

1. identifying the potential advantage of combining these two methods from the point of view of developing robust and effective robots
2. understanding the role of the interaction between learning and evolution in nature

Within an evolutionary perspective, learning has several different adaptive functions. First, it might allow individuals to adapt to changes that occur too quickly to be tracked by evolution [61.95]. Secondly, learning might allow robots to use information extracted during their interaction with environment to develop adaptive characters ontogenetically without necessarily discovering these characters through genetic variations and without encoding these characters in their genome. To understand the importance of this aspect, we should consider that evolutionary adaptation is based on an explicit but concise indication of how well an individual robot coped with its environment – the fitness value of a robot. Ontogenetic adaptation, on the contrary, is based on extremely rich information – the state of the sensors while the robot interacts with its environment. This huge amount of information encodes very indirectly how well an individual is doing in different phases of its lifetime or how it should modify its behavior to increase its fitness. However, evolving robots that have acquired a predisposition to exploit this information to produce adaptive changes during their lifetime might be able to

develop adaptive characteristics on the fly, thus leading to the possibility to produce *complex* phenotypes on the basis of parsimonious genotypes. Finally, learning can help and guide evolution. Although physical changes of the phenotype, such as strengthening of synapses during learning, cannot be written back into the genotype, *Baldwin* [61.96] and *Waddington* [61.97] suggested that learning might indeed affect the evolutionary course in subtle but effective ways. Baldwin's argument was that learning accelerates evolution because suboptimal individuals can reproduce by acquiring during life necessary features for survival. However, variation occurring during successive generation might lead to the discovery of genetic traits that lead to the establishment of the same characteristics that were previously acquired through lifetime learning. This latter aspect of Baldwin's effect, namely indirect genetic assimilation of learned traits, has been later supported by scientific evidence and defined by *Waddington* [61.97] as a canalization effect.

Learning however, also has costs such as: (1) a delay in the ability to acquire fitness (due to the need to develop fit behavior ontogenetically), and (2) increased unreliability due to the fact that the possibility to develop certain abilities ontogenetically is subjected to partially unpredictable characteristics of the robot–environment interaction [61.98]. In the next two subsections we describe two experiments that show some of the potential advantages of combining evolution and learning.

61.7.1 Learning to Adapt to Fast Environmental Variations

Consider the case of a Khepera robot that should explore an arena surrounded by black or white walls to reach a target placed in a randomly selected location [61.95]. Evolving robots are provided with eight sensory neurons that encode the state of the four corresponding infrared sensors and two motor neurons that control the desired speed of the two wheels. Since the color of the walls change every generation and since the color significantly affects the intensity of the response of the infrared sensors, evolving robots should develop an ability to *infer* whether they are currently located in an environment with white or black walls and learn to modify their behavior during lifetime. That is, robots should avoid walls only when the infrared sensors are almost fully activated in the case of arenas with white walls, while they should avoid walls even when the infrared sensors are slightly activated in the case of arenas with black walls.

Robots were provided with a neural controller (Fig. 61.22) including four sensory neurons that encoded

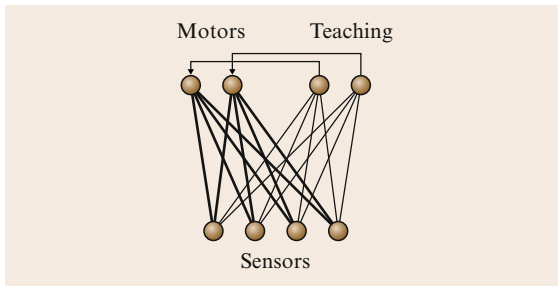


Fig. 61.22 A self-teaching network. The output of the two teaching neurons is used as a teaching value for the two motor neurons. The weights that connect the sensory neurons to the teaching neurons do not vary during the robots' lifetime while the weights that connect the sensory neurons to the motor neurons are modified with an error-correction algorithm

the state of four corresponding infrared sensors; two motors neurons that encoded the desired speed of the two wheels; and two teaching neurons that encoded the teaching values used to modify the connection weights from the sensory neurons to the motor neurons during the robots' lifetime. This special architecture allows evolving robots to transform the sensory states experienced by the robots during their lifetime into teaching signals that might potentially lead to adaptive variations during lifetime. Analysis of evolved robots revealed that they developed two different behaviors that are adapted to the particular arena where they happen to be *born* (surrounded by white or black walls). Evolving robots did not inherit an ability to behave effectively, but rather a predisposition to learn to behave. This predisposition to learn involves several aspects such as a tendency to experience useful learning experiences, a tendency to acquire useful adaptive characters through learning, and a tendency to channel variations toward different directions in different environmental conditions [61.95].

61.7.2 Evolution of Learning

In the previous example, the evolutionary neural network learned using a standard learning rule that was applied to all synaptic connections. *Floreano* and collaborators [61.99] explored the possibility of genetically encoding and evolving the learning rules associated to the different synaptic connections of a neural network embedded in a real robot. The main motivation of this line of work was to evolve robots capable of adapting to a partially unknown environment, rather than robots adapted to the environment(s) seen during evolution. In

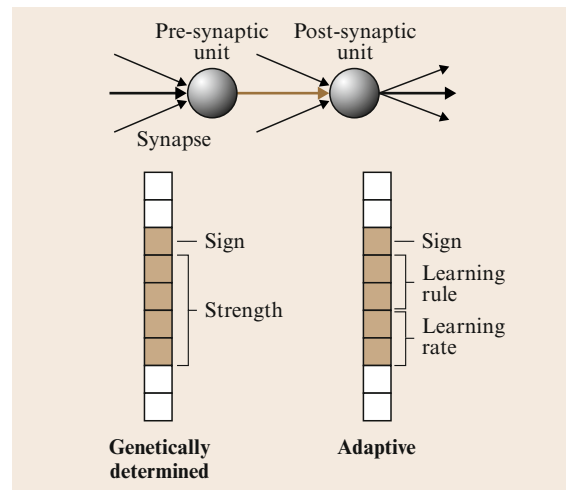


Fig. 61.23 Two methods for genetically encoding a synaptic connection. Genetically determined synapses cannot change during the lifetime of the robot. Adaptive synapses instead are randomly initialized and can change during lifetime of the robot according to the learning rules and rates specified in the genome

order to prevent evolutionary tuning of the neural network to the specificities of the evolutionary environment (which would limit transfer to different environments or transfer from simulation to reality), the synaptic weight values were not genetically encoded. Instead, each synaptic connection in the network was described by three genes that defined its sign, its learning rule, and its learning rate (Fig. 61.23). Every time a genome was decoded into a neural network and downloaded onto the robot, the synaptic strengths were initialized to small random values and could change according to the genetically specified rules and rates while the robot interacted with the environment. Variations of this methodology included a more compact genetic encoding where the learning properties were associated to a neuron instead of a synapse. All synapses afferent to a neuron used its genetically specified rules and rates. Genes could encode four types of Hebbian learning that were modeled upon neurophysiological data and were complementary to each other [61.100].

Experimental results in a nontrivial, multitask environment (Fig. 61.24) indicated that this methodology has a number of significant advantages with respect to the evolution of synaptic strengths without learning [61.45]. Robots evolved faster and obtained better fitness values. Furthermore, evolved behaviors were qualitatively different, notably in that they did not exploit minimal

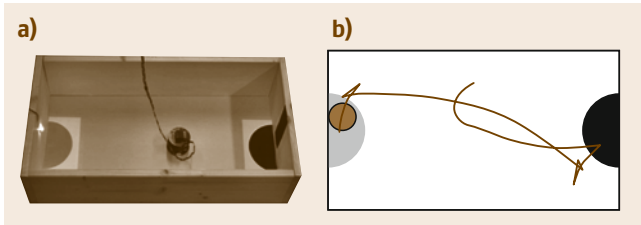


Fig. 61.24 (a) A mobile robot Khepera equipped with a vision module can gain fitness points by staying on the gray area only when the light is on. The light is normally off, but it can be switched on if the robot passes over the black area positioned on the other side of the arena. The robot can detect ambient light and wall color, but not the color of the floor. (b) Behavior of an individual evolved in simulation with genetic encoding of learning rules

solutions tuned to the environment (such as turning only on one side, or turning in circles tuned to the dimensions of the evolutionary arena). Most important, these robots displayed remarkable adaptive properties after evolution. Best evolved individuals: (1) transferred perfectly from simulated to physical robots, (2) accomplished the

task when the light and reflection properties of the environment were modified, (3) accomplished the task when key landmarks and target areas of the environment were displaced, and (4) transferred well across morphologically different robotic platforms. In other words, these robots were selected for their ability to solve a partially unknown problem by adapting on the fly, rather than for being a solution to the problem seen during evolution.

In further experiments where the genetic code for each synapse of the network included one gene whose value caused its remaining genes to be interpreted as connection strengths or learning rules and rates, 80% of the synapses *made the choice* of using learning, reinforcing the fact that this genetic strategy has a comparatively stronger adaptive power [61.100]. This methodology could also be used to evolve the morphology of neural controllers where synapses are created at runtime and therefore their strengths cannot be genetically specified [61.101]. Recently, the adaptive properties of this type of adaptive genetic encoding were confirmed also in the context of evolutionary spiking neurons for robot control [61.102].

61.8 Competition and Cooperation

In the previous sections, we limited our analysis to individual behaviors, i. e., to the evolution of robots placed in an environment that does not include other robots. The evolutionary method, however, can also be applied to develop collective behaviors in which evolving robots are placed in an environment that also contains other individual robots and are selected for the ability to display competitive or cooperative behavior.

In this section we briefly review two examples involving competitive and cooperative behaviors. As we see, the evolution of collective behavior is particularly interesting from the point of view of synthesizing progressively more complex behaviors and from the point of view of developing solutions that are robust with respect to environmental variations.

61.8.1 Coevolving Predator and Prey Robots

Competitive coevolution, for example the coevolution of two populations of predator and prey robots that are evolved for the ability to catch prey and to escape predators, respectively, has two characteristics that are particularly interesting from an evolutionary robotics perspective. The first aspect is that the competition

between populations with different interests might spontaneously lead to a sort of incremental evolutionary process where evolving individuals are faced with progressively more complex challenges (although this is not necessarily the case). Indeed, in initial generations the task of the two populations is relatively simple because opponents have simple and poorly developed abilities on average. After a few generations, however, the abilities of the two populations increase and, consequently, the challenges for each population become more difficult. The second aspect consists of the fact that the environment varies across generations because it includes other coevolving individuals. This implies that coevolving individuals should be able to adapt to ever-changing environments and to develop behaviors that are robust with respect to environmental variations.

The potential advantages of competitive coevolution for evolutionary robotics have been demonstrated by a set of experiments conducted by *Floreano* and *Nolfi* [61.94, 103] where two populations of robots were evolved for the ability to catch prey and escape predators, respectively (Fig. 61.25).

The results indicated that both predator and prey robots tended to vary their behavior throughout gen-

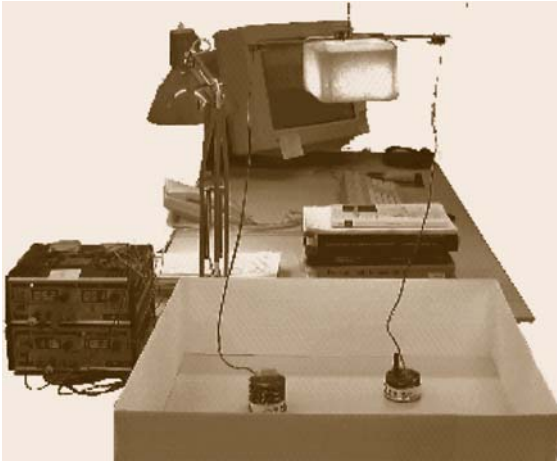


Fig. 61.25 Experimental setup. The predator and prey robot (from left to right) are placed in an arena surrounded by walls and are allowed to interact for several trials starting at different randomly generated orientations. Predators are selected on the basis of the percentage of trials in which they are able to catch (i. e., to touch) the prey, and prey on the basis of the percentage of trials in which they were able to escape (i. e., to not be touched by) predators. Predators have a vision system, whereas the prey have only short-range distance sensors, but can go twice as fast as the predator. Collision between the robots is detected by a conductive belt at the base of the robots

erations without converging on a stable strategy. The behavior displayed by individuals at each generation tended to be tightly adapted to the counter-strategy exhibited by the opponent of the same generation. This evolutionary dynamic however does not really lead to long-lasting progress because, after an initial evolutionary phase, the coevolutionary process led to a limit cycle dynamic where the same small set of behavioral strategies recycled over and over again along generations [61.94]. This limit cycle dynamic can be explained by considering that prey robots tended to vary their behavior in order to disorient predators as soon as predators become effective against the current behavioral strategies exhibited by prey robots.

However, experiments [61.103] where robots were allowed to change their behavior on the fly on the basis of unsupervised Hebbian learning rules showed that the evolutionary phase where coevolving robots were able to produce real progress was significantly longer, and evolved predators displayed an ability to effectively cope with prey exhibiting different behavioral strategies by adapting their behavior on the fly to the prey's behavior.

Prey instead tended to display behavior that changed in unpredictable ways.

Further experiments showed that competitive coevolution can solve problem that the evolution of a single population cannot. *Nolfi and Floreano* [61.94] demonstrated that the attempt to evolve predators robot for the ability to catch a fixed pre-evolved prey produced lower performance with respect to control experiments where predators and prey were coevolved at the same time.

61.8.2 Evolving Cooperative Behavior

Cooperative behavior refers to the situation where a group of robots sharing the same environment coordinate and help each other to solve a problem that cannot be solved by a single robot [61.104]. Although the synthesis of cooperative robots through evolutionary methods is a rather recent enterprise, obtained results are very promising. When it comes to evolving a population of robots for cooperative behaviors, it is necessary to decide the genetic relation among members of a team and the method for selective reproduction. Robots in a team can be genetically homogeneous (clones) or heterogeneous (they differ from each other). Furthermore, the fitness can be computed at the level of the team (in which case, the entire team of robots is reproduced) or at the level of the individual (in which case, only individuals of the team are selected for reproduction). The combination of two variables, genetic relatedness and level of selection, generates at least four different conditions, with a variety of mixed conditions in between. It has been shown experimentally that the choice of homogeneous teams with team-level selection is the most efficient for generating robots that display altruistic cooperation where individual robots are willing to pay a cost for the benefit of the entire team [61.105, 106].

Recent research showed that teams of evolved robots can:

1. develop robust and effective behavior [61.107–109]
2. display an ability to differentiate their behavior in order to better cooperate [61.107, 109]
3. develop communication capabilities and a shared communication system [61.110, 111]

Here we briefly review one of these experiments where swarm-bots [61.112], i. e., teams of autonomous robots capable of dynamically assembling by physically connecting together, were evolved for displaying coordinated motion, navigation on rough terrains, collective negotiation of obstacles and holes, and dynamical shape reorganization in order to go through narrow passages.

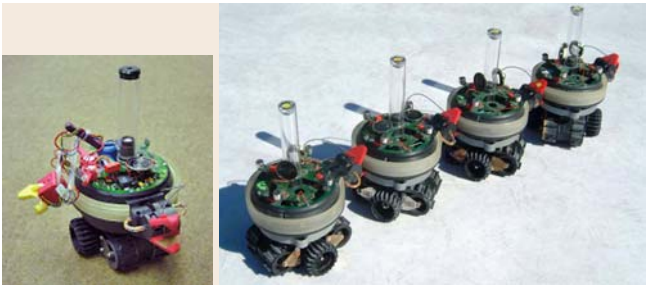


Fig. 61.26 An s-bot and a simulated swarm-bot consisting of four s-bots assembled in chain formation

Each s-bot consisted of a main platform (chassis) and turret that could actively rotate with respect to each other (Fig. 61.26). The chassis included tracks with toothed wheels for navigation on both rough and flat terrain, and four infrared sensors pointing to the ground. The turret included a gripper, a loudspeaker, 16 infrared sensors, three microphones, and a traction sensor placed between

the turret and the chassis to detect the direction and the intensity of the traction force that the turret exerts on the chassis. Each s-bot was provided with a simple neural controller where sensory neurons were directly connected to the motors neurons that controlled the desired speed of the tracks and whether or not a sound signal was produced. The team of s-bots forming a swarm-bot was homogeneous and evolved with team-level selection.

Swarm-bots of four s-bots assembled in chain formation (Fig. 61.26) were evolved for the ability to move coordinately on a flat terrain. Evolved neural controllers were also capable of producing coordinated movements when the swarm-bot was augmented by additional s-bots and reorganized in different shapes. Swarm-bots also dynamically rearranged their shape in order to effectively negotiate narrow passages and were capable of moving on rough terrains by negotiating situations that could not be handled by a single robot. Such robots also collectively avoided obstacles and coordinated to transport heavy objects [61.107, 108, 113].

61.9 Evolutionary Hardware

The work described so far was mainly conducted on robots that did not change shape during evolution, with the exception of self-assembling robots where several

individuals can connect to become a superorganism of variable shape. In recent years, technology advancements have allowed researchers to explore evolution of electronic circuits and morphologies. In this section, we briefly summarize some foundational work in this direction.

61.9.1 Evolvable Hardware Robot Controllers

In most of the work discussed so far some form of genetically specified neural network, implemented in software, has been at the center of the robot control system. Work on a related approach of evolving control systems directly onto hardware dates back to *Thompson's* work in the mid 1990s [61.114]. In contrast to hardware controllers that are designed or programmed to follow a well-defined sequence of instructions, evolved hardware controllers are directly configured by evolution and then allowed to behave in real time according to semiconductor physics. By removing standard electronics design constraints, the physics can be exploited to produce highly nonstandard and often very efficient and minimal systems [61.115].

Thompson [61.114] used artificial evolution to design an onboard hardware controller for a two-wheeled

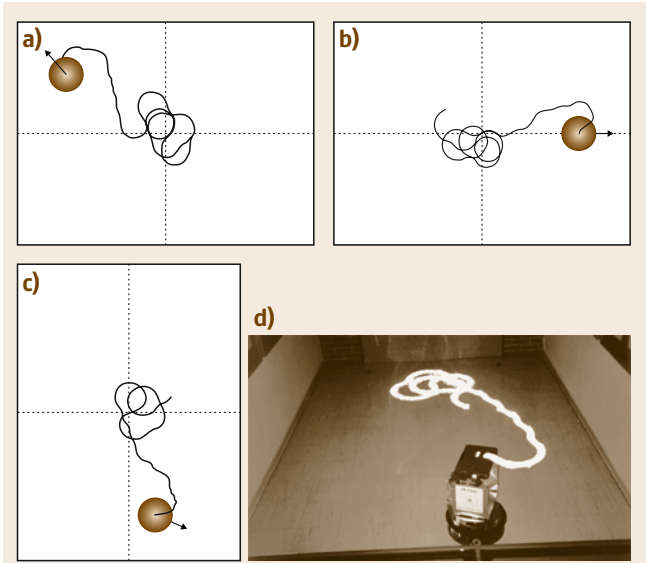


Fig. 61.27a–d Wall-avoidance behavior of a robot with an evolved hardware controller in virtual reality and (d) the real world

autonomous mobile robot engaged in simple wall-avoidance behavior in an empty arena. Starting from a random orientation, and position near the wall, the robot had to move to the center of the arena and stay there using limited sensory input (Fig. 61.27). The DC motors driving the wheels were not allowed to run in reverse and the robot's only sensors were a pair of time-of-flight sonars rigidly mounted on the robot, pointing left and right.

Thompson's approach made use of a so-called dynamic state machine (DSM) – a kind of generalized read-only memory (ROM) implementation of a finite-state machine where the usual constraint of strict synchronization of input signals and state transitions are relaxed (in fact put under evolutionary control). The system had access to a global clock whose frequency was also under genetic control. Thus evolution determined whether each signal was synchronized to the clock or allowed to flow asynchronously. This allowed the evolving DSM to be tightly coupled to the dynamics of interaction between the robot and environment and for evolution to explore a wide range of systems dynamics. The process took place within the robot in a kind of *virtual reality* in the sense that the real evolving hardware controlled the real motors, but the wheels were just spinning in the air. The movements that the robot would have actually performed if the wheels had been supporting it were then simulated and the sonar echo signals that the robot was expected to receive were supplied in real time to the hardware DSM. Excellent performance was attained after 35 generations, with good transfer from the virtual environment to the real world (Fig. 61.27).

Shortly after this research was performed, particular types of field programmable gate arrays (FPGAs) which were appropriate for evolutionary applications became available. FPGAs are reconfigurable systems allowing the construction of circuits built from basic logic elements. Thompson exploited their properties to demonstrate evolution directly in the chip. By again relaxing standard constraints, such as synchronizing all elements with a central clock, he was able to develop very novel forms of functional circuits, including a controller for a Khepera robot using infrared sensors to avoid obstacles [61.115, 116].

Following Thompson's pioneering work, *Keymeulen* evolved a robot control system using a Boolean function approach implemented on gate-level evolvable hardware [61.117]. This system acted as a navigation system for a mobile robot capable of locating and reaching a colored ball while avoiding obstacles. The robot was equipped with infrared sensors and an vision system

giving the direction and distance to the target. A programmable logic device (PLD) was used to implement a Boolean function in its disjunctive form. This work demonstrated that such gate-level evolvable hardware was able to take advantage of the correlations in the input states and to exhibit useful generalization abilities, thus allowing the evolution of robust behavior in simulation followed by a good transfer into the real world.

In a rather different approach, *Ritter* et al. used an FPGA implementation of an onboard evolutionary algorithm to develop a controller for a hexapod robot [61.118]. *Floreano* and collaborators devised a multicellular reconfigurable circuit capable of evolution, self-repair, and adaptation [61.119], and used it as a substrate for evolving spiking controllers of a wheeled robot [61.120]. Although evolved hardware controllers are not widely used in evolutionary robotics, they still hold out the promise of some very useful properties, such as robustness to faults, which make them interesting for extreme condition applications such as space robotics.

61.9.2 Evolving Bodies

In the work described so far there has been an overwhelming tendency to evolve control system for pre-existing robots: the brain is constrained to fit a particular body and set of sensors. Of course in nature the nervous system evolved simultaneously with the rest of the organism. As a result, the nervous system is highly integrated with the sensory apparatus and the rest of the body: the whole operates in a harmonious and balanced way – there are no distinct boundaries between control system, sensors and body. From the start, work at Sussex University incorporated the evolution of sensor properties, including positions, but other aspects of the physical robot were fixed [61.15]. Although the limitations of not being able to control body morphology genetically were acknowledged at this stage, there were severe technical difficulties in overcoming them, so this issue was somewhat sidelined.

Karl Sims started to unlock these possibilities in his highly imaginative work on evolving simulated 3-D creatures in an environment with realistic physics [61.121]. In this work, the creatures coevolved under a competitive scenario in which they were required to try and gain control of a resource (a cube) placed in the center of an arena. Both the morphology of the creatures and the neural system controlling their actuators were under evolutionary control. Their bodies were built from rigid 3-D primitives with the overall morphology being

determined by a developmental process encoded as a directed graph. Various kinds of genetically determined joints were allowed between body parts. A variety of sensors could be specified for a specific body part. The simulated world included realistic modeling of gravity, frictions, collisions, and other dynamics such that behaviors were restricted to be physically plausible. Many different styles of locomotion evolved along with a variety of interesting, and often entertaining, strategies to capture the resource, including pushing the opponent away and covering up the cube.

With the later developments of sophisticated physics engines for modeling a variety of physical bodies, Sims' work inspired a rash of evolved virtual creatures, including realistic humanoid figures capable of a variety of behaviors [61.30].

In what might be regarded as a step towards evolving robot bodies, *Funes* and *Pollack* explored the use of evolutionary algorithms in the design of physical structures taking account of stresses and torques [61.122]. They experimented with evolving structures assembled from elementary components (LEGO bricks). Evolution took place in simulation and the designs were verified in the real world. Stable 3-D brick structures such as tables, cranes, bridges, and scaffolds were evolved within the restrictions of maximum stress torques at each joint between brick pairs. Each brick was modeled as exerting an external load with a lever arm from its center of mass to the supporting joint, resulting in a network of masses and forces representing the structure. A genetic programming approach was taken using tree structures to represent the 3-D LEGO structures. A mutation operator acted on individual brick parameters while subtree crossover allowed more radical changes to the structure. As well as fitness functions designed to encourage particular types of structures, an additional low-level fitness factor favoring the fewest bricks successfully weeded out many of the redundant bricks that inevitably arose. LEGO proved to be a predictable building tool with modes of breakage and linkage that could be relatively easily modeled. While this work was successful, producing very strong designs, it focused on static structures, so was limited in terms of its relevance to functional robotic body parts. However, it did demonstrate a viable approach to evolving physical structures.

While various researchers advocated the use of fully evolvable hardware to develop not only a robot's control circuits, but also its body plan, which might include the types, numbers, and positions of the sensors, the body size, the wheel radius, actuator properties and so on [61.123], this was still largely confined to theoretical

discussion until *Lipson* and *Pollack*'s work on the Golem project [61.124], which was a significant step on from the earlier LEGO work [61.122].

Lipson and Pollack, working at Brandeis University, pushed the idea of fully evolvable robot hardware about as far as is reasonably technologically feasible at present. In an important piece of research, directly inspired by Sims' earlier simulation work, autonomous *creatures* were evolved in simulation out of basic building blocks (neurons, plastic bars, actuators) [61.124]. The bars could connect together to form arbitrary truss structures with the possibility of both rigid and articulated substructures. Neurons could be connected to each other and to bars whose length they would then control via a linear actuator. Machines defined in this way were required to move as far as possible in a limited time. The fittest individuals were then fabricated robotically using rapid manufacturing technology (plastic extrusion 3-D printing) to produce results such as that shown in Fig. 61.28). They thus achieved autonomy of design and construction using evolution in a limited-universe physical simulation coupled to automatic fabrication. The fitness function employed was simply the Euclidean distance moved by the center of mass of a machine over a fixed small number of cycles of its neural controller. A number of different mutation operators acted in concert: small changes to bar or neuron properties, additions and deletions of bars or neurons, changes to connections between neurons and bars, and the creation of new vertices. The highly unconventional designs thus performed as well in reality as in simulation. The success of this work points the way to new possibilities in developing energy-efficient fault-tolerant machines.

Pfeifer and colleagues at Zurich University have explored issues central to the key motivation for fully



Fig. 61.28 A locomoting *creature* evolved by Lipson and Pollack in research which achieved an autonomous design and fabrication process

evolvable robot hardware: the balanced interplay between body morphology, neural processing, and the generation of adaptive behavior [61.125, 126]. Work from this group has involved a number of approaches, including evolutionary robotics. One focus is on studying how to exploit the dynamic interaction between the embodied agent and the environment. Pfeifer uses the term *morphological computation* to draw attention to the fact that some of the control of behavior can be

performed by the dynamic interaction derived from morphological properties (e.g., the passive forward swing of the leg in walking and the spring-like properties of muscles). By taking morphological computation into account, it is very likely that more robust, energy efficient robots can be developed. Pfeifer has developed a set of design principles for intelligent systems that take into account this balance between environment, morphology, and behavior [61.125].

61.10 Closing Remarks

Evolutionary robotics is a young and integrated approach to robot development without human intervention where machines change and adapt by capitalizing on the interactions with their environment. Despite initial skepticism by mainstream and applied robotics practitioners and even by pioneers of this approach [61.127], over the years the field has been constantly growing with new methods and approaches for evolving more complex, efficient, and sometimes surprising robotic systems. In some areas, such as morphology and self-

assembly, evolutionary robotics is still the most widely used and powerful approach.

Evolutionary robotics is not only a method for automatic robot development inspired by biology, but also a tool for investigating open questions in biology concerning evolutionary, developmental, and brain dynamics. Its richness and fecundity make us believe that this approach will continue to grow and progress towards the creation of a new species of machines capable of self-evolution.

References

- 61.1 S. Nolfi, D. Floreano: *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines* (MIT Press/Bradford, Cambridge 2000)
- 61.2 J.H. Holland: *Adaptation in Natural and Artificial Systems* (Univ. of Michigan Press, Ann Arbor 1975)
- 61.3 A.M. Turing: Computing machinery and intelligence, *Mind* **LIX** 236, 433–460 (1950)
- 61.4 V. Braitenberg: *Vehicles. Experiments in Synthetic Psychology* (MIT Press, Cambridge 1984)
- 61.5 R.D. Beer: *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology* (Academic, San Diego 1990)
- 61.6 D. Parisi, F. Cecconi, S. Nolfi: Econets: Neural networks that learn in an environment, *Network* **1**, 149–168 (1990)
- 61.7 P. Husbands, I. Harvey: Evolution versus design: Controlling autonomous robots, Integrating Perception Plan. Action, Proc. 3-rd IEEE Annu. Conf. Artif. Intell. Simul. Plan. (IEEE Press, 1992) pp. 139–146
- 61.8 D. Floreano, O. Miglino, D. Parisi: Emergent complex behaviors in ecosystems of neural networks. In: *Parallel Architectures and Neural Networks*, ed. by E. Caianiello (World Scientific, Singapore 1991)
- 61.9 R.A. Brooks: Intelligence without representation, *Artif. Intell.* **47**, 139–159 (1991)
- 61.10 F. Mondada, E. Franzi, P. lenne: Mobile robot miniaturization: A tool for investigation in control algorithms, Proc. 3-rd Int. Symp. Exp. Robot., ed. by T. Yoshikawa, F. Miyazaki (Springer, Tokyo 1993) pp. 501–513
- 61.11 L. Steels (Ed.): *The Biology and Technology of Intelligent Autonomous Agents*, NATO ASI Series (Springer, Berlin 1995)
- 61.12 D. Floreano, F. Mondada: Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In: *From Animals to Animats 3: Proc. 3-rd Int. Conf. Simul. Adapt. Behav.*, ed. by D. Cliff, P. Husbands, J.A. Meyer, S.W. Wilson (MIT Press–Bradford, Cambridge 1994) pp. 402–410
- 61.13 I. Harvey, P. Husbands, D.T. Cliff: Seeing the light: Artificial evolution, real vision. In: *From Animals to Animats 3: Proc. 3-rd Int. Conf. Simul. Adapt. Behav. SAB94*, ed. by D.T. Cliff, P. Husbands, J.-A. Meyer, S. Wilson (MIT Press, Cambridge 1994) pp. 392–401
- 61.14 M.A. Lewis, A.H. Fagg, A. Solidum: Genetic programming approach to the construction of a neural network for a walking robot, Proc. IEEE Int. Conf. Robot. Autom. (IEEE Press, 1992) pp. 2618–2623

- 61.15 D. Cliff, I. Harvey, P. Husbands: Explorations in evolutionary robotics, *Adapt. Behav.* **2**, 73–110 (1993)
- 61.16 D.E. Goldberg: *Genetic Algorithms in Search, Optimization and Machine Learning* (Addison-Wesley, Redwood City 1989)
- 61.17 R.D. Beer, H.J. Chiel, L.S. Sterling: Heterogeneous neural networks for adaptive behavior in dynamic environments. In: *Neural Information Processing Systems*, Vol. 1, ed. by D. Touretzky (Morgan Kaufman, San Mateo 1989) pp. 577–585
- 61.18 H. de Garis: Genetic programming: Evolution of time dependent neural network modules which teach a pair of stick legs to walk, *Proc. 9th Eur. Conf. Artif. Intell.* (Stockholm 1990) pp. 204–206
- 61.19 R.D. Beer, J.C. Gallagher: Evolving dynamical neural networks for adaptive behavior, *Adapt. Behav.* **1**, 94–110 (1992)
- 61.20 M.A. Lewis, A.H. Fagg, G. Bekey: Genetic algorithms for gait synthesis in a hexapod robot. In: *Recent Trends in Mobile Robots*, ed. by Y. Zheng (World Scientific, New Jersey 1994) pp. 317–331
- 61.21 J. Gallagher, R. Beer, M. Espenschiell, R. Quinn: Application of evolved locomotion controllers to a hexapod robot, *Robot. Auton. Syst.* **19**(1), 95–103 (1996)
- 61.22 R.D. Beer, R.D. Quinn, H.J. Chiel, R.E. Ritzmann: Biologically inspired approaches to robotics, *Commun. ACM* **40**, 31–38 (1997)
- 61.23 S. Galt, B.L. Luk, A.A. Collie: Evolution of smooth and efficient walking motions for an 8-legged robot, *Proc. 6-th Eur. Workshop Learn. Robots* (Brighton 1997)
- 61.24 T. Gomi, K. Ide: Emergence of gaits of a legged robot by collaboration through evolution, *IEEE World Congr. Comput. Intell.* (IEEE Press, New York 1998)
- 61.25 F. Gruau: Automatic definition of modular neural networks, *Adapt. Behav.* **3**(2), 151–183 (1995)
- 61.26 F. Gruau, K. Quatramaran: Cellular encoding for interactive evolutionary robotics, *Proc. 4-th Eur. Conf. Artif. Life*, ed. by P. Husbands, I. Harvey (MIT Press, Cambridge 1997) pp. 368–377
- 61.27 J. Kodjabachian, J.A. Meyer: Evolution and development of neural networks controlling locomotion, gradient following and obstacle avoidance in artificial insects, *IEEE Trans. Neural Netw.* **9**, 796–812 (1998)
- 61.28 N. Jakobi: Running across the reality gap: Octopod locomotion evolved in a minimal simulation. In: *Evolutionary Robotics: 1-st European Workshop, EvoRobot98*, ed. by P. Husbands, J.A. Meyer (Springer, Heidelberg 1998) pp. 39–58
- 61.29 R. T  llez, C. Angulo, D. Pardo: Evolving the walking behavior of a 12 DOF quadruped using a distributed neural architecture, *Proc. 2-nd Int. Workshop Biol. Inspired Approaches to Adv. Inf. Technol. (Bio-ADIT'2006)*, LNCS, Vol. 3853 (Springer, 2006) pp. 5–19
- 61.30 T. Reil, P. Husbands: Evolution of central pattern generators for bipedal walking in real-time physics environments, *IEEE Trans. Evol. Comput.* **6**(2), 10–21 (2002)
- 61.31 www.naturalmotion.com
- 61.32 B. von Haller, A.J. Ijspeert, D. Floreano: Co-evolution of structures and controllers for Neobot underwater modular robots, 8-th European Conference on Artificial Life (ECAL'2005), ed. by M.S. Capcarrere, A.A. Freitas, P.J. Bentley, C.G. Johnson, J. Timmis (Springer, Berlin 2005) pp. 189–199
- 61.33 E. Vaughan, E.A. Di Paolo, I. Harvey: The evolution of control and adaptation in a 3D powered passive dynamic walker, *Proc. 9-th Int. Conf. Simul. Synth. Living Syst. Artif. Life IX*, ed. by J. Pollack, M. Bedau, P. Husbands, T. Ikegami, R. Watson (MIT Press, Cambridge 2004) pp. 139–145
- 61.34 T. McGeer: Passive walking with knees, *Proc. IEEE Conf. Robot. Autom.*, Vol. 2 (IEEE Press, 1990) pp. 1640–1645
- 61.35 E. Vaughan, E.A. Di Paolo, I. Harvey: The tango of a load balancing biped, *Proc. 7-th Int. Conf. Climbing and Walking Robots (CLAWAR)*, ed. by M. Armada, P. Gonzalez De Santos (Springer, 2004)
- 61.36 G. McHale, P. Husbands: Quadrupedal locomotion: Gasnets, CTRNNs and hybrid CTRNN/PNNs compared, *Proc. 9-th Int. Conf. Simul. Synth. Living Syst. (Alife IX)*, ed. by J. Pollack, M. Bedau, P. Husbands, T. Ikegami, R. Watson (MIT Press, Cambridge 2004) pp. 106–112
- 61.37 G. McHale, P. Husbands: GasNets and other evolvable neural networks applied to bipedal locomotion. In: *From Animals to Animats 8: Proc. 8-th Int. Conf. Simul. Adapt. Behav., SAB'2004*, ed. by S. Schaal (MIT Press, Cambridge 2004) pp. 163–172
- 61.38 J.F. Laszlo, M. van de Panne, E. Fiume: Limit cycle control and its application to the animation of balancing and walking, *Proc. SIGGRAPH'96*, Vol. 30 (ACM, 1996) pp. 155–162
- 61.39 R.A. Brooks: Artificial life and real robots. In: *Toward a Practice of Autonomous Systems: Proc. 1-st Eur. Conf. Artif. Life*, ed. by F.J. Varela, P. Bourgine (MIT Press, Cambridge 1992) pp. 3–10
- 61.40 R. Featherstone, D. Orin: Robot dynamics: Equations and algorithms, *Proc. IEEE Int. Conf. Robot. Autom.* (IEEE Press 2000) pp. 826–834
- 61.41 N. Jakobi, P. Husbands, I. Harvey: Noise and the reality gap: The use of simulation in evolutionary robotics. In: *Advances in Artificial Life: Proc. 3-rd Eur. Conf. Artif. Life*, Lecture Notes in Artificial Intelligence, Vol. 929, ed. by F. Moran, A. Moreno, J.J. Merelo, P. Chacon (Springer, Berlin 1995) pp. 704–720

- 61.42 O. Miglino, H.H. Lund, S. Nolfi: Evolving mobile robots in simulated and real environments, *Artif. Life* **2**, 417–434 (1996)
- 61.43 N. Jakobi: Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics, *Proceedings of the 4th European Conference on Artificial Life*, ed. by P. Husbands, I. Harvey (MIT Press, Cambridge 1997) pp. 348–357
- 61.44 J.C. Bongard, H. Lipson: Nonlinear system identification using coevolution of models and tests, *IEEE Trans. Evol. Comput.* **9**(4), 361–384 (2005)
- 61.45 J. Urzelai, D. Floreano: Evolution of adaptive synapses: Robots with fast adaptive behavior in new environments, *Evol. Comput.* **9**, 495–524 (2001)
- 61.46 H.R. Maturana, F.J. Varela: *Autopoiesis and Cognition: The Realization of the Living* (Reidel, Dordrecht 1980)
- 61.47 R.D. Beer: A dynamical systems perspective on agent–environment interaction, *Artif. Intell.* **72**, 173–215 (1995)
- 61.48 P. Funes, B. Orme, E. Bonabeau: Evolving emergent group behaviors for simple humans agents, *Proc. 7-th Eur. Conf. Artif. Life*, ed. by J. Dittrich, T. Kim (Springer, Berlin 2003) pp. 76–89
- 61.49 S. Nolfi: Behaviour as a complex adaptive system: On the role of self-organization in the development of individual and collective behavior, *ComplexUs* **2**, 195–203 (2006)
- 61.50 S. Nolfi: Power and limits of reactive agents, *Neurocomputing* **42**, 119–145 (2002)
- 61.51 I. Harvey, P. Husbands, D.T. Cliff, A. Thompson, N. Jakobi: Evolutionary robotics: The Sussex approach, *Robot. Auton. Syst.* **20**, 205–224 (1997)
- 61.52 P. Husbands, I. Harvey, D. Cliff, G. Miller: Artificial evolution: A new path for AI?, *Brain Cogn.* **34**, 130–159 (1997)
- 61.53 N. Jakobi: Evolutionary robotics and the radical envelope of noise hypothesis, *Adapt. Behav.* **6**, 325–368 (1998)
- 61.54 K.O. Stanley, R. Miikkulainen: Evolving neural networks through augmenting topologies, *Evol. Comput.* **10**(2), 99–127 (2002)
- 61.55 M.A. Arbib: Self-reproducing automata – some implications for theoretical biology. In: *Towards a Theoretical Biology*, Vol. 2, ed. by C.H. Waddington (Edinburgh Univ. Press, Edinburgh 1969) pp. 204–226
- 61.56 J. Aloimonos, I. Weiss, A. Bandopadhyay: Active vision, *Int. J. Comput. Vis.* **1**(4), 333–356 (1987)
- 61.57 R. Bajcsy: Active perception, *Proc. IEEE*, Vol. 76 (1988) pp. 996–1005
- 61.58 D.H. Ballard: Animate vision, *Artif. Intell.* **48**(1), 57–86 (1991)
- 61.59 P.J. Hancock, R.J. Baddeley, L.S. Smith: The principal components of natural images, *Network* **3**, 61–70 (1992)
- 61.60 D. Floreano, T. Kato, D. Marocco, E. Sauser: Coevolution of active vision and feature selection, *Biol. Cybern.* **90**(3), 218–228 (2004)
- 61.61 D. Floreano, M. Suzuki, C. Mattiussi: Active vision and receptive field development in evolutionary robots, *Evol. Comput.* **13**(4), 527–544 (2005)
- 61.62 T.D. Sanger: Optimal unsupervised learning in a single-layer feedforward neural network, *Neural Netw.* **2**, 459–473 (1989)
- 61.63 R. Held, A. Hein: Movement-produced stimulation in the development of visually guided behavior, *J. Comp. Physiol. Psychol.* **56**(5), 872–876 (1963)
- 61.64 R. Held: Plasticity in sensory-motor systems, *Sci. Am.* **213**(5), 84–94 (1965)
- 61.65 I. Harvey, E.A. Di Paolo, R. Wood, M. Quinn, E. Tuci: Evolutionary robotics: A new scientific tool for studying cognition, *Artif. Life* **11**(1–2), 79–98 (2005)
- 61.66 D.T. Cliff: Computational neuroethology: a provisional manifesto. In: *From Animals to Animals: Proc. 1-st Int. Conf. Simul. Adapt. Behav.*, ed. by J.-A. Meyer, S.W. Wilson (MIT Press–Bradford, Cambridge 1991) pp. 29–39
- 61.67 M. Suzuki, D. Floreano, E.A. Di Paolo: The contribution of active body movement to visual development in evolutionary robots, *Neural Netw.* **18**(5/6), 656–665 (2005)
- 61.68 S. Healy (Ed.): *Spatial Representations in Animals* (Oxford Univ. Press, Oxford 1998)
- 61.69 N.A. Schmajuk, H.T. Blair: Place learning and the dynamics of spatial navigation: A neural network approach, *Adapt. Behav.* **1**, 353–385 (1993)
- 61.70 N. Burgess, J.G. Donnett, K.J. Jeffery, J. O’Keefe: Robotic and neuronal simulation of the hippocampus and rat navigation, *Philos. Trans. R. Soc.* **B-352**, 1535–1543 (1997)
- 61.71 J. O’Keefe, L. Nadel: *The Hippocampus as a Cognitive Map* (Clarendon, Oxford 1978)
- 61.72 J.S. Taube, R.U. Muller, J.B. Jr. Ranck: Head-direction cells recorded from the postsubiculum in freely moving rats. I. Description and quantitative analysis, *J. Neurosci.* **10**, 420–435 (1990)
- 61.73 D.E. Rumelhart, J. McClelland, PDP Group: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations* (MIT Press–Bradford, Cambridge 1986)
- 61.74 W. Maas, C.M. Bishop (Ed.): *Pulsed Neural Networks* (MIT Press, Cambridge 1999)
- 61.75 F. Rieke, D. Warland, R. van Steveninck, W. Bialek: *Spikes: Exploring the Neural Code* (MIT Press, Cambridge 1997)
- 61.76 G. Indiveri, P. Verschure: Autonomous vehicle guidance using analog VLSI neuromorphic sensors, *Proc. 7-th Int. Conf. Neural Netw.*, ed. by W. Gerstner, A. Germond, M. Hasler, J.D. Nicoud (Springer, Berlin 1997) pp. 811–816
- 61.77 M.A. Lewis, R. Etienne-Cummings, A.H. Cohen, M. Hartmann: Toward biomorphic control using

- custom aVLSI CPG chips, Proc. IEEE Int. Conf. Robot. Autom. (IEEE Press, 2000) pp. 494–500
- 61.78 D. Floreano, C. Mattiussi: Evolution of spiking neural controllers for autonomous vision-based robots. In: *Evolutionary Robotics. From Intelligent Robotics to Artificial Life*, ed. by T. Gomi (Springer, Tokyo 2001) pp. 38–61
- 61.79 W. Gerstner, J.L. van Hemmen, J.D. Cowan: What matters in neuronal locking?, Neural Comput. **8**, 1653–1676 (1996)
- 61.80 D. Floreano, Y. Epars, J.C. Zufferey, C. Mattiussi: Evolution of spiking neural circuits in autonomous mobile robots, Int. J. Int. Syst. **21**(9), 1005–1024 (2006)
- 61.81 J.A. Gally, P.R. Montague, G.N. Reeke, G.M. Edelman: The NO hypothesis: Possible effects of a short-lived, rapidly diffusible signal in the development and function of the nervous system, Proc. Natl. Acad. Sci. USA, Vol. 87 (1990) pp. 3547–3551
- 61.82 J. Wood, J. Garthwaite: Models of the diffusional spread of nitric oxide: Implications for neural nitric oxide signaling and its pharmacological properties, Neuropharmacology **33**, 1235–1244 (1994)
- 61.83 T.M. Dawson, S.N. Snyder: Gases as biological messengers: Nitric oxide and carbon monoxide in the brain, J. Neurosci. **14**(9), 5147–5159 (1994)
- 61.84 J. Garthwaite, C.L. Boulton: Nitric oxide signaling in the central nervous system, Annu. Rev. Physiol. **57**, 683–706 (1995)
- 61.85 A.O. Philippides, P. Husbands, M. O'Shea: Four-dimensional neuronal signaling by nitric oxide: A computational analysis, J. Neurosci. **20**(3), 1199–1207 (2000)
- 61.86 C. Hölscher: Nitric oxide, the enigmatic neuronal messenger: Its role in synaptic plasticity, Trends Neurosci. **20**, 298–303 (1997)
- 61.87 P. Husbands, T. Smith, N. Jakobi, M. O'Shea: Better living through chemistry: Evolving GasNets for robot control, Connect. Sci. **10**(4), 185–210 (1998)
- 61.88 T.M.C. Smith, P. Husbands, M. O'Shea: Local evolvability, neutrality, and search difficulty in evolutionary robotics, Biosystems **69**, 223–243 (2003)
- 61.89 A.O. Philippides, P. Husbands, T. Smith, M. O'Shea: Flexible couplings: Diffusing neuromodulators and adaptive robotics, Artif. Life **11**(1–2), 139–160 (2005)
- 61.90 A.O. Philippides, P. Husbands, T. Smith, M. O'Shea: Structure based models of NO diffusion in the nervous system. In: *Computational Neuroscience: a Comprehensive Approach*, ed. by J. Feng (Chapman Hall/CRC, Boca Raton 2004) pp. 97–130
- 61.91 A.O. Philippides, S.R. Ott, P. Husbands, T. Lovick, M. O'Shea: Modeling co-operative volume signaling in a plexus of nitric oxide synthase-expressing neurons, J. Neurosci. **25**(28), 6520–6532 (2005)
- 61.92 D. Barañano, C. Ferris, S. Snyder: A typical neural messenger, Trends Neurosci. **24**(2), 99–106 (2001)
- 61.93 T.M.C. Smith, P. Husbands, A. Philippides, M. O'Shea: Neuronal plasticity and temporal adaptivity: Gasnet robot control networks, Adapt. Behav. **10**(3/4), 161–184 (2002)
- 61.94 S. Nolfi, D. Floreano: Learning and evolution, Auton. Robots **7**, 89–113 (1999)
- 61.95 S. Nolfi, D. Parisi: Learning to adapt to changing environments in evolving neural networks, Adapt. Behav. **1**, 75–98 (1997)
- 61.96 J.M. Baldwin: A new factor in evolution, Am. Naturalist **30**, 441–451 (1896)
- 61.97 C.H. Waddington: Canalization of development and the inheritance of acquired characters, Nature **150**, 563–565 (1942)
- 61.98 G. Mayley: Landscapes, learning costs, and genetic assimilation, Evol. Comput. **4**, 213–234 (1997)
- 61.99 D. Floreano, F. Mondada: Evolution of plastic neurocontrollers for situated agents. In: *From Animals to Animats 4: Proc. 4-th Int. Conf. Simul. Adapt. Behav.*, ed. by P. Maes, M. Mataric, J.A. Meyer, J. Pollack, H. Roitblat, S. Wilson (MIT Press-Bradford, Cambridge 1996) pp. 402–410
- 61.100 D. Floreano, J. Urzelai: Evolutionary robots with online self-organization and behavioral fitness, Neural Netw. **13**, 431–443 (2000)
- 61.101 D. Floreano, J. Urzelai: Neural morphogenesis, synaptic plasticity, and evolution, Theory Biosci. **120**(3–4), 225–240 (2001)
- 61.102 E. Di Paolo: Evolving spike-timing-dependent plasticity for single-trial learning in robots, Philos. Trans. R. Soc. London **A-361**, 2299–2319 (2003)
- 61.103 D. Floreano, S. Nolfi: Adaptive behavior in competing co-evolving species, Proc. 4-th Eur. Conf. Artif. Life, ed. by P. Husbands, I. Harvey (MIT Press, Cambridge 1997) pp. 378–387
- 61.104 Y.U. Cao, A.S. Fukunaga, A. Kahng: Cooperative mobile robotics: Antecedents and directions, Auton. Robots **4**, 7–27 (1997)
- 61.105 A. Perez-Urbe, D. Floreano, L. Keller: Effects of group composition and level of selection in the evolution of cooperation in artificial ants, Proc. 7-th Eur. Conf. Artif. Life (ECAL'2003) (Springer, Berlin 2003) pp. 128–137
- 61.106 M. Waibel, D. Floreano, S. Magnenat, L. Keller: Division of labour and colony efficiency in social insects: effects of interactions between genetic architecture, colony kin structure and rate of perturbations, Proc. R. Soc. Ser. B, Vol. 273 (2006) pp. 1815–1823
- 61.107 G. Baldassarre, S. Nolfi, D. Parisi: Evolving mobile robots able to display collective behavior, Artif. Life **9**, 255–267 (2003)
- 61.108 G. Baldassarre, D. Parisi, S. Nolfi: Coordination and behavior integration in cooperating simulated robots, From Animals to Animats 8: Proc. 8-th Int.

- Conf. Simul. Adapt. Behav. (MIT Press, Cambridge 2003) pp.385–394
- 61.109 M. Quinn, L. Smith, G. Mayley, P. Husbands: Evolving controllers for a homogeneous system of physical robots: Structured cooperation with minimal sensors, *Philos. Trans. R. Soc. London A* **361**, 2321–2344 (2003)
- 61.110 M. Quinn: Evolving communication without dedicated communication channels, *Proc. 6-th Eur. Conf. Artif. Life*, ed. by J. Kelemen, P. Sosik (Springer, Berlin 2001) pp.357–366
- 61.111 D. Marocco, S. Nolfi: Self-organization of communication in evolving robots, *Proc. 10-th Int. Conf. Artif. Life*, ed. by L. Rocha, L. Yeager, M. Bedau, D. Floreano, R. Goldstone, A. Vespignani (MIT Press, Cambridge 2006) pp.178–184
- 61.112 F. Mondada, G. Pettinaro, A. Guignard, I. Kwee, D. Floreano, J.L. Deneubourg, S. Nolfi, L.M. Gambardella, M. Dorigo: Swarm-bot: A new distributed robotic concept, *Auton. Robots* **17**, 193–221 (2004)
- 61.113 V. Trianni, S. Nolfi, M. Dorigo: Cooperative hole-avoidance in a swarm-bot, *Robot. Auton. Syst.* **54**, 97–103 (2006)
- 61.114 A. Thompson: Evolving electronic robot controllers that exploit hardware resources. In: *Advances in Artificial Life: Proc. 3-rd Eur. Conf. Artif. Life*, Lecture Notes in Artificial Intelligence, Vol. 929, ed. by F. Moran, A. Moreno, J.J. Merelo, P. Chacon (Springer, Berlin 1995) pp.640–656
- 61.115 A. Thompson: *Hardware Evolution: Automatic Design of Electronic Circuits in Reconfigurable Hardware by Artificial Evolution*, Distinguished Dissertation Series (Springer, Berlin 1998)
- 61.116 A. Thompson: Artificial evolution in the physical world. In: *Evolutionary Robotics. From Intelligent Robots to Artificial Life (ER'97)*, ed. by T. Gomi (AAI Books, Kanata 1997) pp.101–125
- 61.117 D. Keymeulen, M. Durantez, M. Konaka, Y. Kuniyoshi, T. Higuchi: An evolutionary robot navigation system using a gate-level evolvable hardware, *Proc. 1-st Int. Conf. Evolvable Syst.: From Biology to Hardware*, LNCS, Vol.1259, ed. by T. Higuchi (Springer, 1996) pp.195–209
- 61.118 G. Ritter, J.-M. Puiatti, E. Sanchez: Leonardo and discipulus simplex: An autonomous, evolvable six-legged walking robot. In: *Parallel and Distributed Processing*, LNCS, Vol.1586, ed. by J. Rolim (Springer, Puerto Rico 1999) pp.688–696
- 61.119 D. Roggen, D. Floreano, C. Mattiussi: A morphogenetic evolutionary system: Phylogenesis of the POETIC circuit, *Proceedings of the Fifth International Conference on Evolvable Systems*, ed. by A.M. Tyrrell, P.C. Haddow, J. Torresen (Springer, Heidelberg 2003) pp.83–99
- 61.120 D. Roggen, S. Hofmann, Y. Thoma, D. Floreano: Hardware spiking neural network with run-time reconfigurable connectivity in an autonomous robot. In: *NASA/DoD Conference on Evolvable Hardware*, ed. by J. Lohn, R. Zebulum, J. Steincamp, D. Keymeulen, A. Stoica, M.I. Ferguson (IEEE Press, Los Alamitos 2003) pp.189–198
- 61.121 K. Sims: Evolving 3D morphology and behavior by competition, *Proc. Artif. Life IV*, ed. by R.A. Brooks, P. Maes (MIT Press, 1994) pp.28–39
- 61.122 P. Funes, J. Pollack: Evolutionary body building: adaptive physical designs for robots, *Artif. Life* **4**(4), 337–357 (1998)
- 61.123 H.H. Lund, J. Hallam, W.P. Lee: Evolving robot morphology, *Proc. IEEE 4-th Int. Conf. Evol. Comput.* (IEEE Press, 1997) pp.197–202
- 61.124 H. Lipson, J. Pollack: Automatic design and manufacture of robotic lifeforms, *Nature* **406**, 974–978 (2000)
- 61.125 R. Pfeifer, F. Iida, J. Bongard: New robotics: Design principles for intelligent systems, *Artif. Life* **11**(1–2), 99–120 (2005)
- 61.126 R. Pfeifer, C. Scheier: *Understanding Intelligence* (MIT Press, Cambridge 1999)
- 61.127 M. Matarić, D. Cliff: Challenges in evolving controllers for physical robots, *Robot. Auton. Syst.* **19**(1), 67–83 (1996)

Neurorobotics

62. Neurorobotics: From Vision to Action

Michael A. Arbib, Giorgio Metta, Patrick van der Smagt

The lay view of a robot is a mechanical human, and thus robotics has always been inspired by attempts to emulate biology. In this Chapter, we extend this biological motivation from humans to animals more generally, but with a focus on the central nervous systems rather than the bodies of these creatures. In particular, we investigate the sensorimotor loop in the execution of sophisticated behavior. Some of these sections concentrate on cases where vision provides key sensory data. Neuroethology is the study of the brain mechanisms underlying animal behavior, and Sect. 62.2 exemplifies the lessons it has to offer robotics by looking at optic flow in bees, visually guided behavior in frogs, and navigation in rats, turning then to the coordination of behaviors and the role of attention. Brains are composed of diverse subsystems, many of which are relevant to robotics, but we have chosen just two regions of the mammalian brain for detailed analysis. Section 62.3 presents the cerebellum. While we can plan and execute actions without a cerebellum, the actions are no longer graceful and become uncoordinated. We reveal how a cerebellum can provide a key ingredient in an adaptive control system, tuning parameters both within and between motor schemas. Section 62.4 turns to the mirror system, which provides shared representations which bridge between the execution of an action and the observation of that action when performed by others. We develop a neurobiological model of how learning may forge mirror neurons for hand

62.1	Definitions	1453
62.2	Neuroethological Inspiration	1454
62.2.1	Optic Flow in Bees and Robots	1455
62.2.2	Visually Guided Behavior in Frogs and Robots	1456
62.2.3	Navigation in Rat and Robot	1457
62.2.4	Schemas and Coordinated Control Programs	1459
62.2.5	Saliency and Visual Attention	1461
62.3	The Role of the Cerebellum	1462
62.3.1	The Human Control Loop	1462
62.3.2	Models of Cerebellar Control	1463
62.3.3	Cerebellar Models and Robotics	1466
62.4	The Role of Mirror Systems	1467
62.4.1	Mirror Neurons and the Recognition of Hand Movements ..	1467
62.4.2	A Bayesian View of the Mirror System	1470
62.4.3	Mirror Neurons and Imitation	1473
62.5	Extrodution	1474
62.6	Further Reading	1475
	References	1475

movements, provide a Bayesian view of a robot mirror system, and discuss what must be added to a mirror system to support robot imitation. We conclude by emphasizing that, while neuroscience can inspire novel robotic designs, it is also the case that robots can be used as embodied test beds for the analysis of brain models.

62.1 Definitions

Neurorobotics may be defined as *the design of computational structures for robots inspired by the study of the nervous systems of humans and other animals.*

We note the success of *artificial* neural networks – networks of simple computing elements whose connections change with *experience* – as providing a medium for par-

allel adaptive computation that has seen application in robot vision systems and controllers but here we emphasize neural networks derived from the study of specific *neurobiological* systems. Neurorobotics has a twofold aim: creating better machines which employ the principles of natural neural computation; and using the study of bio-inspired robots to improve understanding of the functioning of the brain. Chapter 60, *Biologically Inspired Robots*, complements our study of *brain design* with work on *body design*, the design of robotic control and actuator systems based on careful study of the relevant biology.

Walter [62.1] described two *biologically inspired* robots, the electromechanical tortoises *Machina speculatrix* and *M. docilis* (though each body has wheels not legs). *M. speculatrix* has a steerable photoelectric cell, which makes it sensitive to light, and an electrical contact, which allows it to respond when it bumps into obstacles. The photoreceptor rotates until a light of moderate intensity is registered, at which time the organism orients itself towards the light and approaches it. However, very bright lights, material obstacles, and steep gradients are repellent to the *tortoise*. The latter stimuli convert the photoamplifier into an oscillator, which causes alternating movements of butting and withdrawal, so that the robot pushes small objects out of its way, goes around heavy ones, and avoids slopes. The *tortoise* has a *hutch*, which contains a bright light. When the machine's batteries are charged, this bright light is repellent. When the batteries are low, the light becomes attractive to the machine and the light continues to exert an attraction until the tortoise enters the hutch, where the machine's circuitry is temporarily turned off until the batteries are recharged, at which time the bright hutch light again exerts a negative tropism. The second robot, *M. docilis* was produced by grafting onto *M. speculatrix* a circuit designed to form conditioned reflexes. In one experiment, Walter connected this circuit to the obstacle-avoiding device in *M. speculatrix*. Training consisted of blowing a whistle just before bumping the shell.

Although Walter's controllers are simple and not based on neural analysis, they do illustrate an attempt to gain inspiration from seeking the simplest mechanisms that will yield an interesting class of biologically

inspired robot behaviors, and then showing how different additional mechanisms yield a variety of enriched behaviors. *Braitenberg's* book [62.2] is very much in this spirit and has entered the canon of neurorobotics. While their work provides a historical background for the studies surveyed here, we instead emphasize studies inspired by the computational neuroscience of the mechanisms serving vision and action in the human and animal brain. We seek lessons from linking behavior to the analysis of the internal workings of the brain (1) at the relatively high level of characterizing the functional roles of specific brain regions (or the functional units of analysis called schemas Sect. 62.2.4), and the behaviors which emerge from the interactions between them, and (2) at the more detailed level of models of neural circuitry linked to the data of neuroanatomy and neurophysiology. There are lessons for neurorobotics to be learned from even finer-scale analysis of the biophysics of individual neurons and the neurochemistry of synaptic plasticity but these are beyond the scope of this chapter (see *Segev* and *London* [62.3] and *Fregnac* [62.4], respectively, for entry points into the relevant computational neuroscience).

The plan of this Chapter is as follows. After some selected examples from computational neuroethology, the computational analysis of neural mechanisms underlying animal behavior, we show how perceptual and motor schemas and visual attention provide the framework for our action-oriented view of perception, and show the relevance of the computational neuroscience to robotic implementations (Sect. 62.2). We then pay particular attention to two systems of the mammalian brain, the cerebellum and its role in tuning and coordinating actions (Sect. 62.3), and the mirror system and its roles in action recognition and imitation (Sect. 62.4). The introduction will then invite readers to explore the many other areas in which neurorobotics offers lessons from neuroscience to the development of novel robot designs. What follows, then, can be seen as a contribution to the continuing dialogue between robot behavior and animal and human behavior in which particular emphasis is placed on the search for the neural underpinnings of vision, visually guided action, and cerebellar control.

62.2 Neuroethological Inspiration

Biological evolution has yielded a staggering variety of creatures, each with brains and bodies adapted to spe-

cific niches. One may thus turn to the neuroethology of specific creatures to gain inspiration for special-purpose

robots. In Sect. 62.2.1, we will see how researchers have studied bees and flies for inspiration for the design of flying robots, but have also learned lessons for the visual control of terrestrial robots. In Sect. 62.2.2, we introduce *Rana computatrix*, an evolving model of visuomotor coordination in frogs and toads. The name *the frog that computes*, was inspired by Walter's *M. speculatrix* and inspired in turn the names of a number of other species of neuroethologically inspired robots, including Beer's [62.5] computational cockroach *Periplaneta computatrix* and Cliff's [62.6] hoverfly *Syrretta computatrix*.

Moreover, we learn not only from the brains of specific creatures but also from comparative analysis of the brains of diverse creatures, looking for homologous mechanisms as computational variants which may then be related to the different ecological niches of the creatures that utilize them. A basic theme of brain evolution is that new functions often emerge through modulation and coordination of existing structures. In other words, to the extent that new circuitry may be identified with the new function, it need not be as a module that computes the function autonomously, but rather as one that can deploy prior resources to achieve the novel functionality. Section 62.2.3 will introduce the role of the rat brain in navigation, while Sect. 62.2.4 will look at the general framework of perceptual schemas motor schemas and coordinated control programs for a high-level view of the neuroscience and neurorobotics of vision and action. Finally, Sect. 62.2.5 will look at the control of visual attention in mammals as a homolog of orienting behavior in frogs and toads. All this sets the stage for our emphasis on the roles of the cerebellum (Sect. 62.3) and mirror systems (Sect. 62.4) in the brains of mammals and their implications for neurorobotics. We stress that the choice of these two systems is conditioned by our own expertise, and that studies of many other brain systems also hold great importance for neurorobotics.

62.2.1 Optic Flow in Bees and Robots

Before we turn to vertebrate brains for much of our inspiration for neurorobotics, we briefly sample the rich literature on insect-inspired research. Among the founding studies in computational neuroethology were a series of reports from the laboratory of Werner Reichardt in Tübingen which linked the delicate anatomy of the fly's brain to the extraction of visual data needed for flight control. More than 40 years ago, Reichardt [62.7] published a model of motion detection inspired by this work that has long been central to discussions of visual motion

in both the neuroscience and robotics literatures. Borst and Dickinson [62.8] provide a recent study of continuing biological research on visual course control in flies. Such work has inspired a large number of robot studies, including those of van der Smagt and Groen [62.9], van der Smagt [62.10] Liu and Usseglio-Viretta [62.11], Ruffier et al. [62.12], and Reiser and Dickinson [62.13].

Here, however, we look in a little more detail at honeybees. Srinivasan, Zhang, and Chahl [62.14] continued the tradition of studying image motion cues in insects by investigating how optic flow (the flow of pattern across the eye induced by motion relative to the environment) is exploited by honeybees to guide locomotion and navigation. They analyzed how bees perform a smooth landing on a flat surface: image velocity is held constant as the surface is approached, thus automatically ensuring that flight speed is close to zero at touchdown. This obviates any need for explicit knowledge of flight speed or height above the ground. This landing strategy was then implemented in a robotic gantry to test its applicability to autonomous airborne vehicles. Barron and Srinivasan [62.15] investigated the extent to which ground speed is affected by headwinds. Honey

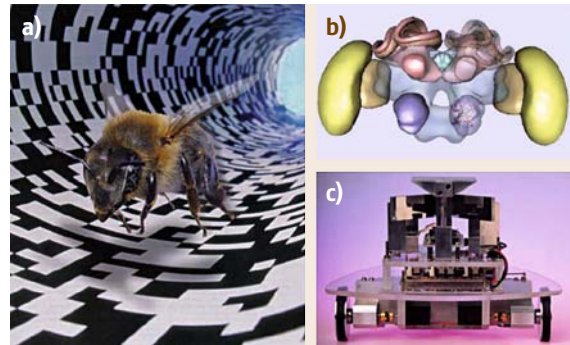


Fig. 62.1a–c Observation of the trajectories of honeybees flying in visually textured tunnels has provided insights into how bees use optic flow cues to regulate flight speed and estimate distance flown, and balance optic flow in the two eyes to fly safely through narrow gaps. This information has been used to build autonomously navigating robots. **(b)** schematic illustration of a honey-bee brain, carrying about a million neurons within about one cubic millimeter. (Images courtesy of M. Srinivasan: **(a)** *Science* **287**, 851–853 (2000); **(b)** *Virtual Atlas of the Honeybee Brain*, <http://www.neurobiologie.fu-berlin.de/beebrain/Bee/VRML/SnapshotCosmoall.jpg>. **(c)** (Research School of Biological Sciences, Australian National University) A mobile robot guided by an optic flow algorithm based on the studies exemplified in **(a)**)

bees were trained to enter a tunnel to forage at a sucrose feeder placed at its far end (Fig. 62.1a). The bees used visual cues to maintain their ground speed by adjusting their airspeed to maintain a constant rate of optic flow, even against headwinds which were, at their strongest, 50% of a bee's maximum recorded forward velocity.

Vladusich et al. [62.16] studied the effect of adding goal-defining landmarks. Bees were trained to forage in an optic-flow-rich tunnel with a landmark positioned directly above the feeder. They searched much more accurately when both odometric and landmark cues were available than when only odometry was available. When the two cue sources were set in conflict, by shifting the position of the landmark in the tunnel during tests, bees overwhelmingly used landmark cues rather than odometry. This, together with other such experiments, suggests that bees can make use of odometric and landmark cues in a more flexible and dynamic way than previously envisaged. In earlier studies of bees flying down a tunnel, Srinivasan and Zhang [62.17] placed different patterns on the left and right walls. They found that bees balance the image velocities in the left and right visual fields. This strategy ensures that bees fly down the middle of the tunnel, without bumping into the side walls, enabling them to negotiate narrow passages or to fly between obstacles. This strategy has been applied to a corridor-following robot (Fig. 62.1c). By holding constant the average image velocity as seen by the two eyes during flight, the bee avoids potential collisions, slowing down when it flies through a narrow passage. The movement-sensitive mechanisms underlying these various behaviors differ qualitatively as well as quantitatively, from those that mediate the optomotor response (e.g., turning to track a pattern of moving stripes) that had been the initial target of investigation of the Reichardt laboratory. The lesson for robot control is that flight appears to be coordinated by a number of visuomotor systems acting in concert, and the same lesson can apply to a whole range of tasks which must convert vision to action. Of course, vision is but one of the sensory systems that play a vital role in insect behavior. Webb [62.18] uses her own work on robot design inspired by the auditory control of behavior in crickets to anchor a far-ranging assessment of the extent to which robotics can offer good models of animal behaviors.

62.2.2 Visually Guided Behavior in Frogs and Robots

Lettvin et al. [62.19] treated the frog's visual system from an ethological perspective, analyzing circuitry in relation

to the animal's ecological niche to show that different cells in the retina and the visual midbrain region known as the tectum were specialized for detecting predators and prey. However, in much visually guided behavior, the animal does not respond to a single stimulus, but rather to some property of the overall configuration. We thus turn to the question *what does the frog's eye tell the frog?*, stressing the embodied nervous system or, perhaps equivalently, an action-oriented view of perception. Consider, for example, the snapping behavior of frogs confronted with one or more fly-like stimuli. Ingle [62.20] found that it is only in a restricted region around the head of a frog that the presence of a fly-like stimulus elicits a snap, that is, the frog turns so that its midline is pointed at the stimulus and then lunges forward and captures the prey with its tongue. There is a larger zone in which the frog merely orients towards the target, and beyond that zone the stimulus elicits no response at all. When confronted with two *flies* within the snapping zone, either of which is vigorous enough that alone it could elicit a snapping response, the frog exhibits one of three reactions: it snaps at one of the flies, it does not snap at all, or it snaps in between at the *average fly*. Didday [62.21] offered a simple model of this choice behavior which may be considered as the prototype for a *winner-take-all* (WTA) model which receives a variety of inputs and (under ideal circumstances) suppresses the representation of all but one of them; the one that remains is the *winner* which will play the decisive role in further processing. This was the beginning of *Rana computatrix* (see Arbib [62.22, 23] for overviews).

Studies on frog brains and behavior inspired the successful use of potential fields for robot navigation strategies. Data on the strategies used by frogs to capture prey while avoiding static obstacles (Collett [62.24]) grounded the model by Arbib and House [62.25] which linked systems for depth perception to the creation of spatial maps of both prey and barriers. In one version of their model, they represented the map of prey by a potential field with long-range attraction and the map of barriers by a potential field with short-range repulsion, and showed that summation of these fields yielded a field that could guide the frog's detour around the barrier to catch its prey. Corbacho and Arbib [62.26] later explored a possible role for learning in this behavior. Their model incorporated learning in the weights between the various potential fields to enable adaptation over trials as observed in the real animals. The success of the models indicated that frogs use reactive strategies to avoid obstacles while moving to a goal, rather than employing a planning or cognitive system. Other work

(e.g., Cobas and Arbib [62.27]) studied how the frog's ability to catch prey and avoid obstacles was integrated with its ability to escape from predators. These models stressed the interaction of the tectum with a variety of other brain regions such as the pretectum (for detecting predators) and the tegmentum (for implementing motor commands for approach or avoidance).

Arkin [62.28] showed how to combine a computer vision system with a frog-inspired potential field controller to create a control system for a mobile robot that could successfully navigate in a fairly structured environment using camera input. The resultant system thus enriched other roughly contemporaneous applications of potential fields in path planning with obstacle avoidance for both manipulators and mobile robots (Khatib [62.29]; Krogh and Thorpe [62.30]). The work on *Rana Computatrix* proceeded at two levels – both biologically realistic neural networks, and in terms of functional units called *schemas*, which compete and cooperate to determine behavior. Section 62.2.4 will show how more general behaviors can emerge from the competition and cooperation of perceptual and motor schemas as well as more abstract coordinating schemas too. Such ideas were, of course, developed independently by a number of authors, and so entered the robotics literature by various routes, of which the best known may be the subsumption architecture of Brooks [62.31] and the ideas of Braitenberg cited above, whereas Arkin's work on behavior-based robotics [62.32] is indeed rooted in schema theory. Arkin et al. [62.33] present a recent example of the continuing interaction between robotics and ethology, offering a novel method for creating high-fidelity models of animal behavior for use in robotic systems based on a behavioral systems approach (i.e., based on a schema-level model of animal behavior, rather than analysis of biological circuits in animal brains), and describe how an ethological model of a domestic dog can be implemented with AIBO, the Sony entertainment robot.

62.2.3 Navigation in Rat and Robot

The tectum, the midbrain visual system which determines how the frog turns its whole body towards it prey or orients it for escape from predators (Sect. 62.2.2), is homologous with the superior colliculus of the mammalian midbrain. The rat superior colliculus has been shown to be *frog-like*, mediating approach and avoidance (Dean et al. [62.34]), whereas the best-studied role of the superior colliculus of cat, monkey, and human is in the control of saccades, rapid eye movements to ac-

quire a visual target. Moreover, the superior colliculus can integrate auditory and somatosensory information into its visual frame (Stein and Meredith [62.35]) and this inspired Strosslin et al. [62.36] to use a biologically inspired approach based on the properties of neurons in the superior colliculus to learn the relation between visual and tactile information in control of a mobile robot platform. More generally, then, the comparative study of mammalian brains has yielded a rich variety of computational models of importance in neurorobotics. In this section, we further introduce the study of *mammalian neurorobotics* by looking at studies of mechanisms of the rat brain for spatial navigation.

The frog's detour behavior is an example of what O'Keefe and Nadel [62.37] called the *taxon* (*behavioral orientation*) system [as in Braitenberg, [62.38] a *taxis* (plural *taxes*) is an organism's response to a stimulus by movement in a particular direction]. They distinguished this from a system for *map-based navigation*, and proposed that the latter resides in the hippocampus, though Guazzelli et al. [62.39] qualified this assertion, showing how the hippocampus may function as *part of* a cognitive map. The taxon versus map distinction is akin to the distinction between reactive and deliberative control in robotics (Arkin et al. [62.33]). It will be useful to relate taxis to the notion of an *affordance* (Gibson [62.40]), a feature of an object or environment relevant to action, for example, in picking up an apple or a ball, the identity of the object may be irrelevant, but the size of the object is crucial. Similarly, if we wish to push a toy car, recognizing the make of car copied in the toy is irrelevant, whereas it is crucial to recognize the placement of the wheels to extract the direction in which the car can be readily pushed. Just as a rat may have basic taxes for approaching food or avoiding a bright light, say, so does it have a wider repertoire of affordances for possible actions associated with the immediate sensing of its environment. Such affordances include *go straight ahead* for visual sighting of a corridor, *hide* for a dark hole, *eat* for food as sensed generically, *drink* similarly, and the various turns afforded by, e.g., the sight of the end of the corridor. It also makes rich use of olfactory cues. In the same way, a robot's behavior will rely on a host of reactions to local conditions in fulfilling a plan, e.g., knowing that it must go to the end of a corridor it will nonetheless use local visual cues to avoid hitting obstacles, or to determine through which angle to turn when reaching a bend in the corridor.

Both normal and hippocampal-lesioned rats can learn to solve a simple T-maze (e.g., learning whether to turn left or right to find food) in the absence of any

consistent environmental cues other than the T-shape of the maze. If anything, the lesioned animals learn this problem faster than normals. After the criterion was reached, probe trials with an eight-arm radial maze were interspersed with the usual T-trials. Animals from both groups consistently chose the side to which they were trained on the T-maze. However, many did not choose the 90° arm but preferred either the 45° or 135° arm, suggesting that the rats eventually solved the T-maze by learning to rotate within an egocentric orientation system at the choice point through approximately 90°. This leads to the hypothesis of an *orientation vector* being stored in the animal's brain but does not tell us where or how the orientation vector is stored. One possible model would employ coarse coding in a linear array of cells, coding for turns from −180° to +180°. From the behavior, one might expect that only the cells close to the preferred *behavioral* direction are excited, and that learning *marches* this peak from the old to the new pre-

ferred direction. To *unlearn* −90°, say, the array must reduce the peak there, while at the same time *building* a new peak at the new direction of +90°. If the old peak has *mass* $p(t)$ and the new peak has *mass* $q(t)$, then as $p(t)$ declines toward 0 while $q(t)$ increases steadily from 0, the center of mass will progress from −90° to +90°, fitting the behavioral data.

The determination of movement direction was modeled by *rat-ification* of the *Arbib* and *House* [62.25] model of frog detour behavior. There, prey was represented by excitation coarsely coded across a population, while barriers were encoded by inhibition whose extent closely matched the retinotopic extent of each barrier. The sum of excitation was passed through a winner-takes-all circuit to yield the choice of movement direction. As a result, the direction of the gap closest to the prey, rather than the direction of the prey itself, was often chosen for the frog's initial movement. The same model serves for behavioral orientation once

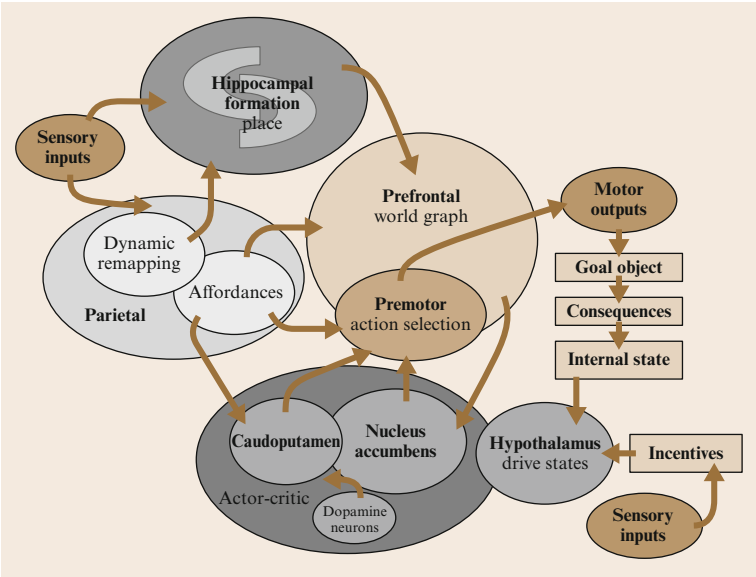


Fig. 62.2 The TAM-WG model has at its basis a system, TAM (the taxon affordance model), for exploiting affordances. This is elaborated by a system, WG (the world graph), which can use a cognitive map to plan paths to targets which are not currently visible. Note that the model processes two different kinds of sensory inputs. At the *bottom right* are those associated with, e.g., hypothalamic systems for feeding and drinking, and that may provide both incentives and rewards for the animal's behavior, contributing both to behavioral choices, and to the reinforcement of certain patterns of behavior. The nucleus accumbens and caudo-putamen mediate an actor-critic style of reinforcement learning based on the hypothalamic drive of the dopamine system. The sensory inputs at the *top left* are those that allow the animal to sense its relation with the external world, determining both where it is (the hippocampal place system) as well as the affordances for action (the parietal recognition of affordances can shape the premotor selection of an action). The TAM model focuses on the parietal-premotor reaction to immediate affordances; the WG (world graph) model places action selection within the wider context of a cognitive map. (after Guazzelli et al. [62.41])

we replace the direction of the prey (frog) by the direction of the orientation vector (rat), while the barriers correspond to the presence of walls rather than alley ways.

To approach the issue of how a cognitive map can extend the capability of the affordance system, Guazzelli et al. [62.43] extended the Lieblisch and Arbib [62.44] approach to building a cognitive map as a *world graph*, a set of nodes connected by a set of edges, where the nodes represent recognized places or situations, and the links represent ways of moving from one situation to another. A crucial notion is that a place encountered in different circumstances may be represented by multiple nodes, but that these nodes may be merged when the similarity between these circumstances is recognized. They model the process whereby the animal decides where to move next, on the basis of its current drive state (hunger, thirst, fear, etc.). The emphasis is on spatial maps for guiding locomotion into regions not necessarily current visible, rather than retinotopic representations of immediately visible space, and yields exploration and latent learning without the introduction of an explicit exploratory drive. The model shows: (1) how a route, possibly of many steps, may be chosen that leads to the desired goal; (2) how short cuts may be chosen; and (3) through its account of node merging why, in open fields, place cell firing does not seem to depend on direction.

The overall structure and general mode of operation of the complete model is shown in Fig. 62.2, which gives a vivid sense of the lessons to be learned by studying not only specific systems of the mammalian brain but also their patterns of large-scale interaction. This model is but one of many inspired by the data on the role of the hippocampus and other regions in rat navigation. Here, we just mention as pointers to the wider literature the papers by Girard et al. [62.45] and Meyer et al. [62.46], which are part of the *Psikharpax* project, which is doing for rats what *Rana computatrix* did for frogs and toads.

62.2.4 Schemas and Coordinated Control Programs

Schema theory complements neuroscience's well-established terminology for levels of *structural* analysis (brain region, neuron, synapse) with a *functional* vocabulary, a framework for analysis of behavior with no necessary commitment to hypotheses on the localization of each *schema* (unit of functional analysis), but which can be linked to a structural analysis whenever appropriate. Schemas provide a high-level vocabulary which can

be shared by brain theorists, cognitive scientists, connectionists, ethologists, kinesiologists – and roboticists. In particular, schema theory can provide a distributed programming environment for robotics [see, e.g., the robots schemas (RS) language of Lyons and Arbib [62.47], and supporting architectures for distributed control as in Metta et al. [62.48]]. Schema theory becomes specifically relevant to *neurorobotics* when the schemas are inspired by a model constrained by data provided by, e.g., human brain mapping, studies of the effects of brain lesions, or neurophysiology.

A *perceptual schema* not only determines whether an object or other *domain of interaction* is present in the environment but can also provide important parameters to motor schemas (see below) for the guidance of action. The activity level of a perceptual schema signals the credibility of the hypothesis that what the schema represents is indeed present, whereas other schema parameters represent relevant properties such as size, location, and motion of the perceived object. Given a perceptual schema we may need several schema instances, each suitably tuned, to subserve perception of several instances of its domain, e.g., several chairs in a room.

Motor schemas provide the control systems which can be coordinated to affect a wide variety of actions.

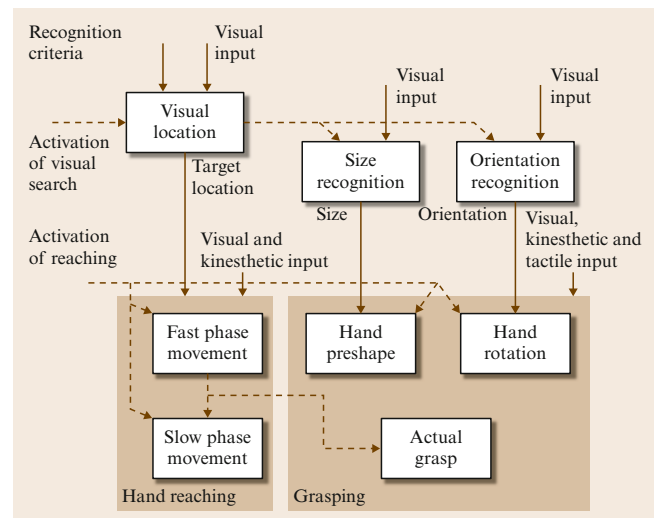


Fig. 62.3 Hypothetical coordinated control program for reaching and grasping. The perceptual schemas (*top*) provide parameters for the motor schemas (*bottom*) for the control of *reaching* (arm transport \approx and reaching) and *grasping* (controlling the hand to conform to the object). *Dashed lines* indicate activation signals which establish timing relations between schemas; *solid lines* indicate transfer of data. (After Arbib [62.42])

The activity level of a motor schema instance may signal its *degree of readiness* to control some course of action. What distinguishes schema theory from usual control theory is the transition from emphasizing a few basic controllers (e.g., for locomotion or arm movement) to a large variety of motor schemas for diverse skills (peeling an apple, climbing a tree, changing a light bulb), with each motor schema depending on perceptual schemas to supply information about objects which are targets for interaction. Note the relevance of this for robotics – the robot needs to know not only *what* the object is but also *how* to interact with it. Modern neuroscience (see the works by Ungerleider and Mishkin [62.49] and Goodale and Milner [62.50]) has indeed established that the monkey and human brain each use a dorsal pathway (via the parietal lobe) for the *how* and a ventral pathway (via the inferotemporal cortex) for the *what*. Moreover, coupling between these two *streams* mediates their integration in normal ongoing behavior.

A *coordinated control program* interweaves the activation of various perceptual, motor, and coordinating schemas in accordance with the current task and sensory environment to mediate complex behaviors. Figure 62.3 shows the original coordinated control program (Arbib [62.42], inspired by the data of Jeannerod and Biguer [62.51]). As the hand moves to grasp a ball, it is *preshaped* so that, when it has almost reached the ball, it is of the right shape and orientation to enclose some part of the ball prior to gripping it firmly. The outputs of three perceptual schemas are available for the concurrent activation of two motor schemas, one controlling the arm to transport the hand towards the object and the other preshaping the hand. Once the hand is preshaped, it is only the completion of the fast initial phase of hand transport that *wakes up* the final stage of the grasping schema to shape the fingers under control of tactile feedback. [This model anticipates the much later discovery of perceptual

schemas for grasping in a localized area (AIP) of parietal cortex and motor schemas for grasping in a localized area (F5) of premotor cortex; see Fig. 62.4.] The notion of schema is thus *recursive* – a schema may be analyzed as a coordinated control program of finer schemas, and so on until such time as a secure foundation of neural specificity is attained.

Subsequent work has refined the scheme of Fig. 62.3, for example, Hoff and Arbib’s [62.52] model uses the *time needed for completion* of each of the movements – transporting the hand and preshaping the hand – to explain data on how the reach to grasp responds to perturbation of target location or size. Moreover, Hoff and Arbib [62.53] show how to embed an optimality principle for arm trajectories into a controller which can use feedback to resist noise and compensate for target perturbations, and a predictor element to compensate for delays from the periphery. The result is a feedback system which can *act like* a feedforward system described by the optimality principle in *familiar* situations, where the conditions of the desired behavior are not perturbed and accuracy requirements are such that *normal* errors in execution may be ignored. However, when perturbations must be corrected for or when great precision is required, feedback plays a crucial role in keeping the behavior close to that desired, taking account of delays in putting feedback into effect. This integrated view of feedback and feedforward within a single motor schema seems to us of value for neurorobotics as well as the neuroscience of motor control.

It is standard to distinguish a *forward* or *direct* model which represents the path from motor command to motor output, from the *inverse* model which models the reverse pathway, i.e., going from a desired motor outcome to a set of motor commands likely to achieve it. As we have just suggested, the action plan unfolds *as if* it were feedforward or open-loop when the actual parameters of

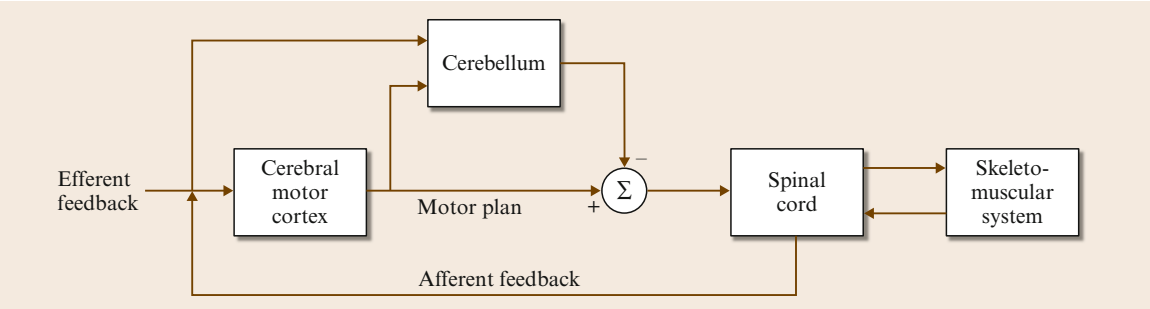


Fig. 62.4 Simplified control loop relating cerebellum and cerebral motor cortex in supervising the spinal cord’s control of the skeleto-muscular system

the situation match the stored parameters, while a feedback component is employed to counteract disturbances (current feedback) and to learn from mistakes (learning from feedback). This is obtained by relying on a forward model that predicts the outcome of the action as it unfolds in real time. The accuracy of the forward model can be evaluated by comparing the output generated by the system with the signals derived from sensory feedback (Miall et al. [62.54]). Also, delays must be accounted for to address the different propagation times of the neural pathways carrying the predicted and actual outcome of the action. Note that the forward model in this case is relatively simple, predicting only the motor output in advance: since motor commands are generated internally it is easy to imagine a predictor for these signals (known as an efference copy). The inverse model, on the other hand, is much more complicated since it maps sensory feedback (e.g., vision) back into motor terms. These concepts will prove important both in our study of the cerebellum (Sect. 62.3) and mirror systems (Sect. 62.4).

62.2.5 Saliency and Visual Attention

Discussions of how an animal (or robot) grasps an object assume that the animal or robot is attending to the relevant object. Thus, whatever the subtlety of processing in the canonical and mirror systems for grasping, its success rests on the availability of a visual system coupled to an oculomotor control system that bring foveal vision to bear on objects to set parameters needed for successful interaction. Indeed, the general point is that attention greatly reduces the processing load for animal and robot. The catch, of course, is that reducing computing load is a Pyrrhic victory unless the moving focus of attention captures those aspects of behavior relevant for the current task – or supports necessary *priority interrupts*. Indeed, directing attention appropriately is a topic for which there is a great richness of both neurophysiological data and robotic application (see Deco and Rolls [62.55] and Choi, et al. [62.41]).

In their neuromorphic model of the bottom-up guidance of attention in primates, Itti and Koch [62.56] decompose the input video stream into eight feature channels at six spatial scales. After surround suppression, only a sparse number of locations remain active in each map, and all maps are combined into a unique *saliency map*. This map is scanned by the focus of attention in order of decreasing saliency through the interaction between a winner-takes-all mechanism (which selects the most salient location) and an *inhibition-of-return* mechanism (which transiently sup-

presses recently attended locations from the saliency map). Because it includes a detailed low-level vision front-end, the model has been applied not only to laboratory stimuli, but also to a wide variety of natural scenes, predicting a wealth of data from psychophysical experiments.

When specific objects are searched for, low-level visual processing can be biased both by the gist (e.g., *outdoor suburban scene*) and also for the features of that object. This top-down modulation of bottom-up processing results in an ability to guide search towards targets of interest (Wolfe [62.57]). Task affects eye movements (Yarbus [62.58]), as do training and general expertise. Navalpakkam and Itti [62.59] propose a computational model which emphasizes four aspects that are important in biological vision: determining the task relevance of an entity, biasing attention for the low-level visual features of desired targets, recognizing these targets using the same low-level features, and incrementally building a visual map of task relevance at every scene location. It attends to the most salient location in the scene, and attempts to recognize the attended object through hierarchical matching against object representations stored in long-term memory. It updates its working memory with the task relevance of the recognized entity and updates a topographic task-relevance map with the location and relevance of the recognized entity, for example, in one task the model forms a map of likely locations of cars from a video clip filmed while driving on a highway. Such work illustrates the continuing interaction between models based on visual neurophysiology and human psychophysics with the tackling of practical robotic applications.

Orabona et al. [62.60] implemented an extension of the Itti–Koch model on a humanoid robot with moving eyes, using log-polar vision as in Sandini and Tagliasco [62.61], and changing the feature construction pyramid by considering proto-object elements (blob-like structures rather than edges). The inhibition-of-return mechanism has to take into account a moving frame of reference, the resolution of the fovea is very different from that at the periphery of the visual field, and head and body movements need to be stabilized. The control of movement might thus have a relationship with the structure and development of the attention system. Rizzolatti et al. [62.62] proposed a role for the feedback projections from premotor cortex to the parietal lobe, assuming that they form a tuning signal that dynamically changes visual perception. In practice this can be seen as an implicit attention system which *selects* sensory information while the action is being prepared and subsequently ex-

ecuted (see *Flanagan and Johansson* [62.63], *Flanagan et al.* [62.64], and *Mataric and Pomplun* [62.65]). The early responses, before action onset, of many premo-

tor and parietal neurons suggest a premotor mechanism of attention that deserves exploration in further work in neurorobotics.

62.3 The Role of the Cerebellum

Although cerebellar involvement in muscle control was advocated long ago by the Greek gladiator surgeon Galen of Pergamum (129–216/17 CE), it was the publication by *Eccle et al.* [62.66] of the first comprehensive account of the detailed neurophysiology and anatomy of the cerebellum (*Ito* [62.67]) that provided the inspiration for the Marr–Albus model of cerebellar plasticity (*Marr* [62.68]; *Albus* [62.69]) that is at the heart of most current modeling of the role of the cerebellum in control of motion and sensing. From a robotics point of view, the most convincing results are based on *Albus'* [62.70] cerebellar model articulation controller (CMAC) model and subsequent implementations by *Miller* [62.71]. These models, however, are only remotely based on the structure of the biological cerebellum. More detailed models are usually only applied to two-degree-of-freedom robotic structures, and have not been generalized to real-world applications (see *Peters and van der Smagt* [62.72]). The problem may lie with viewing the cerebellum as a stand-alone dynamics controller. An important observation about the brain is that schemas are widely distributed, and different aspects of the schemas are computed in different parts of the brain. Thus, one view is that (1) the cerebral cortex has the necessary models for choosing appropriate actions and getting the general shape of the trajectory assembled to fit the present context, whereas (2) the cerebellum provides a side-path which (on the basis of extensive learning of a forward motor model) provides the appropriate corrections to compensate for control delays, muscle nonlinearities, Coriolis and centrifugal forces occasioned by joint interactions, and subtle adjustments of motor neuron firing in simultaneously active motor pattern generators to ensure their smooth coordination. Thus, for example, a patient with cerebellar lesions may be able to move his arm to successfully reach a target, and to successfully adjust his hand to the size of an object. However, he lacks the machinery to perform either action both swiftly and accurately, and further lacks the ability to coordinate the timing of the two subactions. His behavior will thus exhibit *decomposition of movement* – he may first move the hand till the thumb touches the object, and only then shape the hand appropriately to

grasp the object. Thus analysis of how various components of cerebral cortex interact to support forward and inverse models which determine the *overall shape of the behavior* must be complemented by analysis of how the cerebellum handles control delays and nonlinearities to transform a well-articulated plan into graceful coordinated action. Within this perspective, cerebellar structure and function will be very helpful in the control of a new class of highly antagonistic robotic systems as well as in adaptive control.

62.3.1 The Human Control Loop

Lesions and deficits of the cerebellum impair the coordination and timing of movements while introducing excessive, undesired motion: effects which cannot be compensated by the cerebral cortex. According to mainstream models, the cerebellum filters descending motor cortex commands to cope with timing issues and communication delays which go up to 50 ms one way for arm control. Clearly, closed-loop control with such delays is not viable in any reasonable setting, unless augmented with an open-loop component, predicting the behavior of the actuator system. This is where the cerebellum comes into its own. The complexity of the vertebrate musculoskeletal system, clearly demonstrated by the human arm using a total of 19 muscle groups for planar motion of the elbow and shoulder alone (see *Nijhof and Kouwenhoven* [62.73]) requires a control mechanism coping with this complexity, especially in a setting with long control delays. One cause for this complexity is that animal muscles come in antagonistic pairs (e.g., flexing versus extending a joint). Antagonistic control of muscle groups leads to energy-optimal (*Damsgaard et al.* [62.74]) and intrinsically flexible systems. Contact with stiff or fast-moving objects requires such flexibility to prevent breakage. In contrast, classical (industrial) robots are stiff, with limb segments controlled by linear or rotary motors with gear boxes. Even so, most laboratory robotic systems have passively stiff joints, with active joint flexibility obtainable only by using fast control loops and joint torque measurement. Although it may be debatable whether such robotic systems re-

quire cerebellar-based controllers, the steady move of robotics towards complete anthropomorphism by mimicking human (hand and arm) kinematics as well as dynamics as closely as possible, requires the search for alternative, neuromorphic control solutions.

Vertebrate motor control involves the cerebral motor cortex, basal ganglia, thalamus, cerebellum, brain stem, and spinal cord. Motor programs, originating in the cortex, are fed into the cerebellum. Combined with sensory information through the spinal cord, it sends motor commands out to the muscles via the brain stem and spinal cord, which controls muscle length and joint stiffness (see *Bullock and Contreras-Vidal* [62.75]). The full control loop is depicted in Fig. 62.4 (see *Schaal and Schweighofer* [62.76], for an overview of robotic versus brain control loops). The model in Fig. 62.4 clearly resembles the well-known computed torque model and, when the cerebellum is interpreted as a Smith model, it serves to cope with long delays in the control loop (see *Miall et al.* [62.54] and *van der Smagt and Hirzinger* [62.77]). It is thus understood to incorporate a forward model of the skeletomuscular system.

Alternative approaches use the cerebellum as an inverse model (see *Ebadzadeh et al.* [62.78]), which however leads to increased complexity and control loop stability problems.

62.3.2 Models of Cerebellar Control

The cerebellum can be divided into two parts: the cortex and the deep nuclei. There are two systems of fibers bringing input to the both the cortex and nuclei: the mossy fibers and the climbing fibers. The only output from the cerebellar nucleus comes from cells called Purkinje cells, and they project only to the cerebellar nuclei, where their effect is inhibitory. This inhibition sculpts the output of the nuclei which (the effect varies from nucleus to nucleus) may act by modulating activity in the spinal cord, the mid-brain or the cerebral cortex. We now turn to models which make explicit use of the cellular structure of the *cerebellar cortex* (see *Eccles et al.* [62.66] and *Ito* [62.79], and also Fig. 62.5a). The human cerebellum has 7–14 million Purkinje cells (PCs), each receiving about 200 000 synapses. Mossy

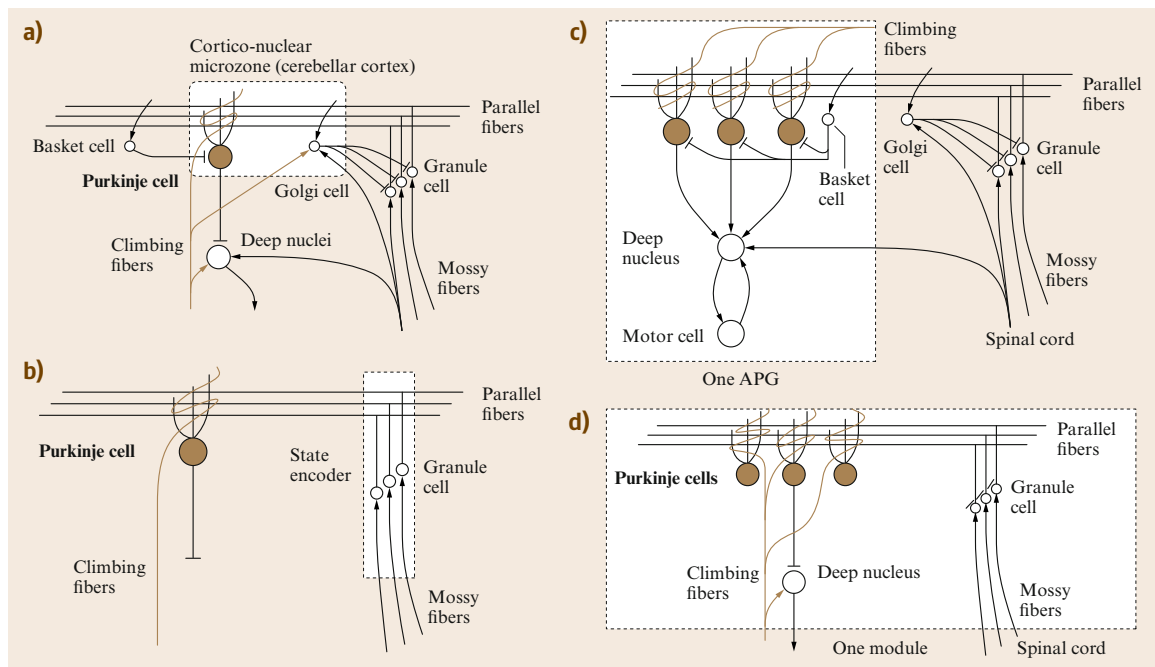


Fig. 62.5 (a) Major cells in the cerebellum. (b) Cells in the Marr–Albus model. The granule cells are state encoders, feeding system state, and sensor data into the PC. PC/PF synapses are adjusted using the Widrow–Hoff rule. The output of the PC are steering signals for the robotic system. (c) The APG model, using the same state encoder as in (b). (d) The MPMF model. A single module corresponds to a group of Purkinje cells: predictor, controller, and responsibility estimator. The granule cells generate the necessary basis functions of the original information

fibers (MFs) arise from the spinal cord and brainstem. They synapse onto granule cells and deep cerebellar nuclei. Granule cells have axons which each project up to form a T, with the bars of the T forming the parallel fibers (PFs). Each PF synapses on about 200 PCs. The PCs, which are grouped into microzones, inhibit the deep nuclei. PCs with their target cells in cerebellar nuclei are grouped together in microcomplexes (see Ito [62.79]). Microcomplexes are defined by a variety of criteria to serve as the units of analysis of cerebellar influence on specific types of motor activity. The climbing fibers (CF) arise from the inferior olive. Each PC receives synapses from only one CF, but a CF makes about 300 excitatory synapses on each PC which it contacts. This powerful input alone is enough to fire the PC, though most PC firing depends on subtle patterns of PF activity. The cerebellar cortex also contains a variety of inhibitory interneurons. The basket cell is activated by PF afferents and makes inhibitory synapses onto PCs. Golgi cells receive input from PFs, MFs, and CFs and inhibit granule cells.

The Marr–Albus Model

In the Marr–Albus model (see Marr [62.68] and Albus [62.69]) the cerebellum functions as a classifier of sensory and motor patterns received through the MFs. Only a small fraction of the parallel fibers (PF) are active when a Purkinje cell (PC) fires and thus influence the motor neurons. Both Marr and Albus hypothesized that the error signals for improving PC firing in response to PF, and thus MF input, were provided by the climbing fibers (CF), since only one CF affects a given PC. However, Marr hypothesized that CF activity would strengthen the active PF/PC synapses using a Widrow–Hoff learning rule, whereas Albus hypothesized they would weaken them. This is an important example of a case where computational modeling inspired important experimentation. Eventually, Masao Ito was able to demonstrate that Albus was correct – the weakening of active synapses is now known to involve a process called long-term depression (Ito [62.79]). However, the rule with weakening of synapses still known as the Marr–Albus model, and remains the reference model for studies of synaptic plasticity of cerebellar cortex. However, both Marr and Albus viewed each PC as functioning as a perceptron whose job it was to control an elemental movement, contrasting with more plausible models in which PCs serve to modulate the involvement of microcomplexes (which include cells of the deep nuclei) in motor pattern generators (e.g., the APG model described below).

Since the development of the Marr–Albus model several cerebellar models have been introduced in which cerebellar plasticity plays a key role. Limiting our overview to computational models, we will describe (1) the cerebellar model articulation controller (CMAC), (2) the adjustable pattern generator (APG), (3) the Schweighofer–Arbib model, and (4) the multiple paired forward-inverse models (see van der Smagt [62.80,81]).

The Cerebellar Model Articulation Controller (CMAC)

One of the first well-known computational models of the cerebellum is the CMAC (Albus [62.70]; see Fig. 62.5b). The algorithm was based on Albus' understanding of the cerebellum, but it was not proposed as a biologically plausible model. The idea has its origins in the BOXES approach, in which for n variables an n -dimensional hypercube stores function values in a lookup table. BOXES suffers from the curse of dimensionality: if each variable can be discretized into D different steps, the hypercube has to store D^n function values in memory. Albus assumed that the mossy fibers provided discretized function values. If the signal on a mossy fiber is in the receptive field of a particular granule cell, it fires onto a parallel fiber. This mapping of inputs onto binary output variables is often considered to be the generalization mechanism in CMAC. The learning signals are provided by the climbing fibers.

Albus' CMAC can be described in terms of a large set of overlapping, multidimensional receptive fields with finite boundaries. Every input vector falls within the range of some local receptive fields. The response of CMAC to a given input is determined by the average of the responses of the receptive fields excited by that input. Similarly, the training for a given input vector affects only the parameters of the excited receptive fields.

The organization of the receptive fields of a typical Albus CMAC with a two-dimensional input space can be described as follows. The set of overlapping receptive fields is divided into C subsets, commonly referred to as layers. Any input vector excites one receptive field from each layer, for a total of C excited receptive fields for any input. The overlap of the receptive fields produces input generalization, while the offset of the adjacent layers of receptive fields produces input quantization. The ratio of the width of each receptive field (input generalization) to the offset between adjacent layers of receptive fields (input quantization) must be equal to C for all dimensions of the input space. This organization of the receptive fields guarantees that only a fixed number, C , of receptive fields is excited by any input.

If a receptive field is excited, its response equals the magnitude of a single adjustable weight specific to that receptive field. The **CMAC** output is the average of the weights of the excited receptive fields. If nearby points in the input space excite the same receptive fields, they produce the same output value. The output only changes when the input crosses one of the receptive field boundaries. The Albus **CMAC** thus produces piecewise-constant outputs. Learning takes place as described above.

CMAC neural networks have been applied in various control situations *Miller* [62.71], starting from adaptation of **PID** control parameters for an industrial robot arm and hand-eye systems up to biped walking (see *Sabourin and Bruneau* [62.82]).

The Adjustable Pattern Generator APG

The **APG** model (*Houk et al.* [62.83]) got its name because the model can generate a burst command with adjustable intensity and duration. The **APG** is based on the same understanding of the mossy fiber–granule cell–parallel fiber structure as **CMAC**, using the same state encoder, but has the crucial difference (Fig. 62.2c) that the role of the nuclei is crucial. In the **APG** model, each nucleus cell is connected to a motor cell in a feedback circuit. Activity in the loop is then modulated by Purkinje cell inhibition, a modeling idea introduced by *Arbib* et al. [62.84].

The learning algorithm determines which of the **PF–PC** synapses will be updated in order to improve movement generation performance. This is the traditional credit assignment problem: *which* synapse (the structural credit assignment) must be updated based on a response issued *when* (temporal credit assignment). While the former is solved by the **CFs**, which are considered binary signals, for the latter eligibility traces on the synapses are introduced, serving as memory for recent activity to determine which synapses are eligible for updates. The motivation for the eligibility signal is this: each firing of a **PC** cell will take some time to affect the animal's movement, and a further delay will occur before the **CF** can signal an error in the movement in which the **PC** is involved. Thus the error signal should not affect those **PF–PC** synapses which are currently active, but should instead act upon those synapses which affected the activity whose error is now being registered.

The **APG** has been applied in a few control situations, e.g., a single muscle–mass system and a simulated two-link robot arm. Unfortunately these applications do not allow us judge the performance of the **APG** scheme

itself due to the fact that the control task itself was hidden within spinal cord and muscle models.

The Schweighofer–Arbib Model

The Schweighofer–Arbib model was introduced in *Schweighofer* [62.85]. It does not use the **CMAC** state encoder but tries to copy the anatomy of the cerebellum. All the cells, fibers, and axons in Fig. 62.2a are included. Several assumptions are made: (1) there are two types of mossy fibers, one type reflecting the desired state of the controlled plant and another which carries information on the current state. A mossy fiber diverges into approximately 16 branches; (2) granule cells have an average of four dendrites, each of which receive input from different mossy fibers through a synaptic structure called the glomerulus; (3) three Golgi cells synapse on a granule cell through the glomerulus and the strength of their influence depends on the simulated geometric distance between the glomerulus and the Golgi cell; (4) the climbing fiber connection on nuclear cells as well as deep nuclei is neglected.

Learning in this model depends on directed error information given by the climbing fibers from the inferior olive (IO). Here, long-term depression is performed when the IO firing rate provides an error signal for an eligible synapse, while compensatory but slower increases in synaptic strength can occur when no error signal is present. Schweighofer applied the model to explain several acknowledged cerebellar system functions: (1) saccadic eye movements, (2) two-link limb movement control (see *Schweighofer et al.* [62.86, 87]), and (3) prism adaptation (*Arbib et al.* [62.88]). Furthermore, control of a simulated human arm was demonstrated.

Multiple Paired Forward–Inverse Models (MPFIM)

Building on a long history of cerebellar modeling, *Wolpert and Kawato* [62.89] proposed a novel functional model of the cerebellum which uses multiple coupled predictors and controllers which are trained for control, each being responsible for a small state-space region. The **MPFIM** model is based on the indirect/direct model approach by Kawato, and is also based on the microcomplex theory. We noted earlier that a microzone is a group of **PCs**, while a microcomplex combines the **PCs** of a microzone with their target cells in cerebellar nuclei. In **MPFIM**, a microzone consists of a set of modules controlling the same degree of freedom and is learned by only one particular climbing fiber. The modules in this microzone compete to control this particular synergy. Inside such a module there are three types of **PC** which

perform the computations of a forward model, an inverse model or a responsibility predictor, but all receiving the same input. A single internal model i is considered to be a controller which generates a motor command τ_i and a predictor which predicts the current acceleration. Each predictor is a forward model of the controlled system, while each controller contains an inverse model of the system in a region of specialization. The responsibility signal weights the contribution that this model will make to the overall output of the microzone. Indeed, MPFIM further assumes that each microzone contains n internal models of situations occurring in the control task. Model i generates motor command τ_i , and estimates its own responsibility r_i . The feedforward motor command τ_{ff} consists only of the output of the single models adjusted by the sum of responsibility signals: $\tau_{\text{ff}} = \sum r_i \tau_i / \sum r_i$.

The PCs are considered to be roughly linear. The MF inputs carry all necessary information including state information, efference copies of the last motor commands as well as desired states. Granule cells, and eventually the inhibitory interneurons as well, nonlinearly transform the state information to provide a rich set of basis functions through the PFs. A climbing fiber carries a scalar error signal while each Purkinje cell encodes a scalar output – responsibilities, predictions, and controller outputs are all one-dimensional values. MPFIM has been introduced with different learning methods: its first implementations were done using gradient descent methods; subsequently, expectation maximization (EM) batch-learning and hidden Markov chain EM learning have been applied.

Comparison of the Models

Summing up, we can categorize the cerebellar models CMAC, APG, Schweighofer–Arbib, and MPFIM as follows.

- **State-encoder-driven models:** This kind of model assumes that the granule cells are on–off types of entities which split up the state space. This kind of model is best suited for, e.g., simple function approximation, and suffers strongly from the curse of dimensionality.
- **Cellular-level models:** Obviously, the most realistic simulations would be at the cellular level. Unfortunately, modeling only a few Purkinje cells at realistic conditions is an immense computational challenge, and other relevant neurons are even less well understood. Still, from the biological point of view this kind of model is the most important since it allows

obtaining insight into cerebellar function on cellular level. The first steps in this direction were taken by the Schweighofer–Arbib model.

- **Functional models:** From the computer-science point of view, the most interesting models are based on functional understanding of the cells. In this case, we obtain only a basic insight of the functions of the parts and apply it as a crude approximation. This kind of approach is very promising and MPFIM, with its emphasis on the use of responsibility signals to combine models appropriately, provides an interesting example of this approach.

62.3.3 Cerebellar Models and Robotics

From the previous discussions, it is clear that a popular view is that the function of the cerebellum within the motor control loop is to represent a forward model of the skeletomuscular system. As such it predicts the movements of the body, or rather the perceptually coded (e.g., through muscle spindles, skin-based positional information, and visual feedback) representation of the movements of the body. With this prediction a fast control loop between motor cortex and cerebellum can be realized, and motor programs are played before being sent to the spinal cord (Fig. 62.4). Proprioceptive feedback is used for adaptation of the motor programs as well as for updating the forward model stored in the cerebellum. However, the Schweighofer–Arbib model is based on the view that the cerebellum offers not so much a total forward model of the skeletomuscular system as a forward model of the *difference* between the crude model of the skeletomuscular system available to the motor planning circuits of the cerebral cortex, and the more intricately parameterized forward model of the skeletomuscular system needed to support fast, graceful movements with minimal use of feedback. This hypothesis is reinforced by the fact that cerebellar lesions do not prohibit motion but substantially reduce its quality, since the forward model of the skeletomuscular system is of lesser quality.

As robotic systems move towards their biological counterparts, the control approaches can or must do the same. There are many lines of research investigating the former part; cf. Chap. 13 *Robots with Flexible Arms* and Chap. 60 *Biologically Inspired Robots*. It should be noted that the drive principle that is used to move the joints does not necessarily have a major impact on the outer control loop. Whether McKibben muscles, which are intrinsically flexible but bulky (see *van der Smagt et al.* [62.90]), low-dynamics polymer linear actuators,

or direct-current (DC) motors with spindles and added elastic components are used does not affect the control approach at the cerebellar level, but rather at the motor control level (cf. the spinal cord level). Of key importance, however, are the resulting dynamical properties of the system, which are of course influenced by its actuators.

Passive flexibility at the joints, which is a key feature of muscle systems, is essential for reasons of safety, stability during contact with the environment, and storage of kinetic energy. As mentioned before, however, biological systems are immensely complex, requiring large groups of muscles for comparatively simple movements. A reason for this complexity is the resulting nearly linear behavior, which has been noted for, e.g., muscle activation with respect to joint stiffness (see *Osu* and

Gomi [62.91]). By this regularization of the complexity of the skeletomuscular system, the complexity of the forward model stored in the cerebellum is correspondingly reduced. The whole picture therefore seems to be that the cerebellum, controlling a piecewise-linear skeletomuscular system, incorporates a forward model thereof to cope with delays in the peripheral nervous system. Consequently, although the applicability of cerebellar systems to highly nonlinear dynamics control of traditional robots is questionable, the use of cerebellar systems as forward models appears to be useful in the control of more complex and flexible robotic systems. The control challenge posed by the currently emerging generation of robots employing antagonistic motor control therefore opens a new wealth of applications of cerebellar systems.

62.4 The Role of Mirror Systems

Mirror neurons were first discovered in the brain of the macaque monkey – neurons that fire both when the monkey exhibits a particular grasping action, and when the monkey observes another (monkey or human) perform a similar grasp (see *Rizzolatti* et al. [62.92] and *Gallese* et al. [62.93]). Since then, human studies have revealed a *mirror system* for grasping in the human brain – a mirror system for a class X of actions being a set of regions that are active both when the human performs some action from X and when the human observes someone else performing an action from X (see, e.g., *Grafton* et al. [62.94], *Rizzolatti* et al. [62.95], and *Fadiga* et al. [62.96]). Until recently we had no single-neuron studies of humans proving the reasonable hypothesis that the human mirror system for grasping contains mirror neurons for specific actions. However, data from neurosurgery are now becoming available (M. Iacoboni, personal communication). In any case, most models of the human mirror system for grasping assume that it contains circuitry analogous to the mirror neuron circuitry of the macaque brain. However, the current consensus is that monkeys have little or no ability for imitation (but see *Voelkl* and *Huber* [62.97] for a very simple form of imitation-like behavior in marmoset monkeys); great apes have the ability to master certain skills after extended bouts of observation (*Byrne* [62.98]), whereas the human mirror system plays a key role in our capability for much richer forms of imitation (see *Iacoboni* et al. [62.99]), pantomime, and even language (see *Arbib* and *Rizzo-*

latti [62.100] and *Arbib* [62.101]). Indeed, it has been suggested that mirror neurons underlie the motor theory of speech perception of *Lieberman* et al. [62.102], which holds that speech perception rests on the ability to recognize the motor acts that produce speech sounds.

Section 62.4.1 reviews basic neurophysiological data on mirror neurons in the macaque, and presents both the Fagg–Arbib–Rizzolatti–Sakata (**FARS**) model of canonical neurons (unlike mirror neurons, these are active when the monkey executes an action but not when he observes it) and the mirror neuron system (**MNS**) model of mirror neurons and their supporting brain regions. Section 62.4.2 then uses Bayes’s rule to offer a new, probabilistic view of the mirror system’s role in action recognition, and demonstrates the operation of the new model in the context of studies with two robots. Finally, Sect. 62.4.3 briefly shifts the emphasis of our study of mirror neurons to imitation, which in fact is the area that has most captured the imagination of roboticists.

62.4.1 Mirror Neurons and the Recognition of Hand Movements

Area F5 in the premotor cortex of the macaque contains, among others, neurons which fire when the monkey executes a specific manual action, e.g., one neuron might fire when the monkey performs a precision pinch, another when it executes a power grasp. (In discussing neurorobotics, it seems unnecessary to explain in any detail the areas like F5, **AIP**, and **STS** described here

– they will function as labels for components of functional systems. To fill in the missing details see, e.g., Rizzolatti et al. [62.103, 104]) A subset of these neurons, the so-called *mirror neurons*, also discharge when the monkey observes meaningful hand movements made by the experimenter which are similar to those whose execution is associated with the firing of the neuron. In contrast, the *canonical neurons* are those belonging to the complementary, anatomically segregated subset of grasp-related F5 neurons which fire when the monkey performs a specific action and also when it sees an object as a possible target of such an action – but do not fire when the monkey sees another monkey or human perform the action. Finally, F5 contains a large population of motor neurons which are active when the monkey grasps an object (either with the hand or mouth) but do not possess any visual response. F5 is clearly a motor area although the details of the muscular activation are abstracted out – F5 neurons can be effector-independent. In contrast, the primary motor cortex (F1) formulates the neural instructions for lower motor areas and motor neurons.

Moreover, macaque mirror neurons encode transitive actions and do not fire when the monkey sees the hand movement unless it can also see the object or, more subtly, if the object is not visible but is appropriately *located* in working memory because it has recently been placed on a surface and has then been obscured by a screen be-

hind which the experimenter is seen to be reaching (see Umiltà et al. [62.105]). All mirror neurons show visual generalization. They fire when the instrument of the observed action (usually a hand) is large or small, far from or close to the monkey. They may also fire even when the action instrument has shapes as different as those of a human or monkey hand. Some neurons respond even when the object is grasped by the mouth. When naive monkeys first see small objects grasped with a pair of pliers, mirror neurons do not respond, but after extensive training some precision pinch mirror neurons do show activity also to this new grasp type (see Ferrari et al. [62.106]).

Mirror neurons for grasping have also been found in parietal areas of the macaque brain and, recently, it has been shown that parietal mirror neurons are sensitive to the context of the observed action being predictive of the outcome as a function of contextual cues – e.g., some grasp-related parietal mirror neurons may fire for a grasp that precedes eating the grasped object while others fire for a grasp that precedes placing the object in a container (see Fogassi et al. [62.107]). In practice the parieto-frontal circuitry seems to encode action execution and simultaneously action recognition by taking into account a large set of potential candidate actions which are selected on the basis of a range of cues such as vision of the relation of the effector to the object and certain sounds (when relevant for the task). Further, feedback connections (frontal to parietal) are thought to be part of a stimulus selection process which refines the sensory processing by *attending* to stimuli relevant for the ongoing action (see Rizzolatti et al. [62.62] and recall the discussion in Sect. 62.2.5). Recognition is then supported by the activation of the same circuitry in the absence of overt movement.

We clarify these ideas by briefly presenting the **FARS** model of the canonical F5 neurons and the **MNS** model of the F5 mirror neurons. In each case, the F5 neurons function effectively only because of the interaction of F5 with a wide range of other regions. We have stressed (Sect. 62.2.3) the distinction between recognition of the category of an object and recognition of its *affordances*. The parietal area **AIP** processes visual information to extract *affordances*, in this case properties of the object relevant to grasping it (Taira et al. [62.108]). **AIP** and F5 are reciprocally connected, with **AIP** being more visual and F5 more motoric.

The Fagg–Arbib–Rizzolatti–Sakata (**FARS**) model (see Fagg and Arbib [62.109] and Fig. 62.6) embeds F5 canonical neurons in a larger system. The dorsal stream (which passes through **AIP**) can only analyze the

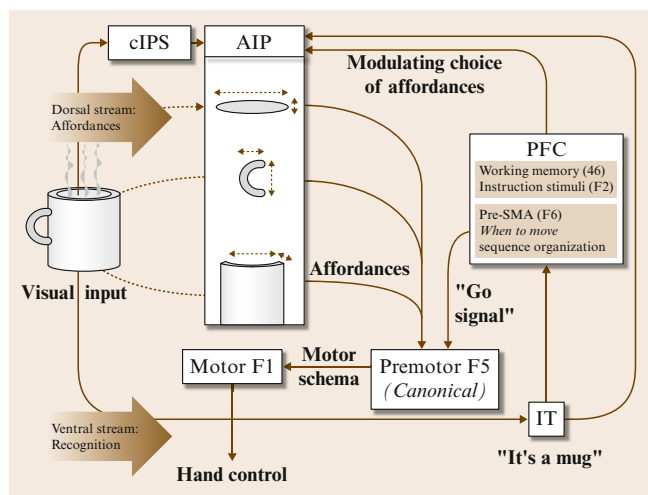


Fig. 62.6 The original **FARS** diagram (see Fagg and Arbib [62.45]) is here modified to show **PFC** acting on **AIP** rather than F5. The idea is that the prefrontal cortex uses the **IT** identification of the object, in concert with task analysis and working memory, to help the **AIP** select the appropriate affordance from its *menu*

object as a set of possible affordances, whereas the ventral stream (via the inferotemporal cortex, **IT**) is able to recognize what the object is. The latter information is passed to the prefrontal cortex (**PFC**) which can then, on the basis of the current goals of the organism, bias the choice of affordances appropriate to the task at hand. Neuroanatomical data (as analyzed by *Rizzolatti and Luppino* [62.110]) suggest that **PFC** and **IT** may modulate action selection at the level of the parietal cortex. Figure 62.6 gives a partial view of the **FARS** model updated to show this modified pathway. The affordance selected by **AIP** activates **F5** neurons to command the appropriate grip once they receive a *go* signal from another region, **F6**, of the prefrontal cortex. **F5** also accepts signals from other **PFC** areas to respond to working memory and instruction stimuli in choosing among the available affordances. Note that this same pathway could be implicated in tool use, bringing in semantic knowledge as well as perceptual attributes to guide the dorsal system (see *Johnson-Frey* [62.111]).

With this, we turn to the *mirror* system. Since grasping a complex object requires careful attention to motion of, e.g., fingertips relative to the object we hold that the primary evolutionary impetus for the mirror system was to facilitate feedback control of dexterous movement. We now show how *parameters relevant to such feedback* could be crucial in enabling the monkey to associate the visual appearance of what it is doing with the task at hand. The key side-effect will be that this feedback-serving self-recognition is so structured as to

also support recognition of the action when performed by others – and it is this *recognition of the actions of others* that has created the greatest interest in mirror neurons and systems.

The **MNS** model of *Oztop and Arbib* [62.112] provides some insight into the anatomy while focusing on the learning capacities of mirror neurons. Here, the task is to determine whether the shape of the hand and its trajectory are *on track* to grasp an observed affordance of an object using a known action. The model is organized around the idea that the **AIP** → **F5_{canonical}** pathway emphasized in the **FARS** model (Fig. 62.6) is complemented by another pathway **7b** → **F5_{mirror}**. As shown in Fig. 62.7 (middle diagonal), object features are processed by **AIP** to extract grasp affordances, these are sent on to the canonical neurons of **F5** that choose a particular grasp. Recognizing the location of the object (top diagonal) provides parameters to the motor programming area **F4** which computes the reach. The information about the reach and the grasp is taken by the motor cortex **M1** (= **F1**) to control the hand and the arm. The rest of the figure provides components that can learn and apply key criteria for activating a mirror neuron, recognizing that the preshape of the observed hand corresponds to the grasp that the mirror neuron encodes and is appropriate to the object, and that the hand is moving on an appropriate trajectory. Making crucial use of input from the superior temporal sulcus (**STS**; see *Perrett et al.* [62.113] and *Carey et al.* [62.114]), schemas at the bottom left recognize the shape of the observed hand,

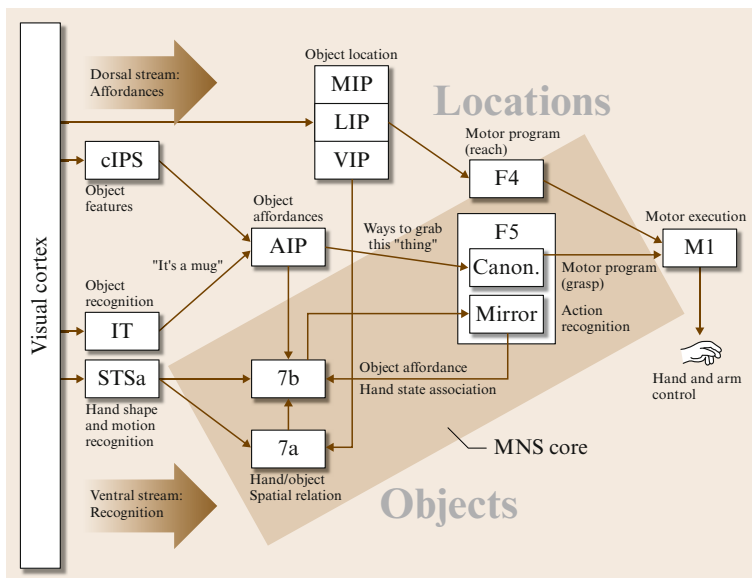


Fig. 62.7 The mirror neuron system (**MNS**) model (see *Oztop and Arbib* [62.92]). Note that this basic mirror system for grasping crucially links the visual process of the superior temporal sulcus (**STS**) to the parietal regions (b) and premotor regions (**F5**) which have been shown to contain mirror neurons for manual actions

and how that hand is moving. Other schemas implement hand–object spatial relation analysis and check how object affordances relate to hand state. Together with F5 canonical neurons, this last schema (in parietal area 7b) provides the input to the F5 mirror neurons.

In the **MNS** model, the *hand state* was defined as a vector whose components represented the movement of the wrist relative to the location of the object and of the hand shape relative to the affordances of the object. Oztop and Arbib showed that an artificial neural network corresponding to **PF** and F5_{mirror} could be trained to recognize the grasp type from the *hand state trajectory*, with correct classification often being achieved well before the hand reached the object, using activity in the F5 canonical neurons that commands a grasp as training signal for recognizing it visually. Crucially, this training prepares the F5 mirror neurons to respond to hand–object relational trajectories even when the hand is of the *other* rather than the *self* because the hand state is based on the view of movement of a hand relative to the object, and thus only *indirectly* on the retinal input of seeing hand and object, which can differ greatly between observation of self and other. Bonaiuto et al. [62.115] have developed MNS2, a new version of the **MNS** model to address data on audiovisual mirror neurons that respond to the sight and sound of actions with characteristic sounds such as paper tearing and nut cracking (see Kohler et al. [62.105]), and on the response of mirror neurons when the target object was recently visible but is currently hidden (see Umiltà et al. [62.105]). Such learning models, and the data they address, make clear that *mirror neurons are not restricted to recognition of an innate set of actions but can be recruited to recognize and encode an expanding repertoire of novel actions*.

The discussion of this section avoided any reference to imitation (Sect. 62.4.3). On the other hand, even without considering imitation, mirror neurons provide a new perspective for tackling the problem of robotic perception by incorporating action (and motor information) into a plausible recognition process. The role of the fronto-parietal system in relating affordances, plans, and actions shows the crucial role of motor information and embodiment. We argue that this holds lessons for neurorobotics: the richness of the *motor* system should strongly influence what the robot can learn, proceeding autonomously via a process of exploration of the environment rather than overly relying on the intermediary of logic-like formalisms. When recognition exploits the ability to act, then the breadth of the action space becomes crucially related to the precision, quality, and robustness of the robot's perception.

62.4.2 A Bayesian View of the Mirror System

We now show how to cast much that is known about the mirror system into a controller–predictor model (see Miall et al. [62.54] and Wolpert et al. [62.116]) and analyze the system in Bayesian terms. As shown by the **FARS** model, the decision to initiate a particular grasping action is attained by the convergence in area F5 of several factors including contextual and object-related information; similarly many factors affect the recognition of an action. All this depends on learning both *direct* (from decision to executed action) and *inverse* models (from observation of an action to activation of a motor command that could yield it). Similar procedures are well known in the computational motor control literature (see Jordan and Rumelhart [62.117] and Kawato et al. [62.118]). Learning of the affordances of objects with respect to grasping can also be achieved autonomously by learning from the consequences of applying many different actions to different parts of different objects.

But how is the decision made to classify an observed behavior as an instance of one action or another? Many comparisons could be performed in parallel with the model for one action becoming predominantly activated. There are plausible implementations of this mechanism using a gating network (see Demiris and Johnson [62.119] and Haruno et al. [62.120]). A gating network learns to partition an input space into regions; for each region a different model can be applied or a set of models can be combined through an appropriate weight function. The design of the gating network can encourage collaboration between models (e.g., linear combination of models) or competition (choosing only one model rather than a combination). Oztop et al. [62.121] offer a similar approach to the estimation of the mental states of the observed actor, using some additional circuitry involving the frontal cortex.

We now offer a Bayesian view of using the predictor–controller formulation approach to the mirror system. This Bayesian approach views affordances as priors in the action recognition process where the evidence is conveyed by the visual information of the hand, providing the data for finding the posterior probabilities as mirror neurons-like responses which automatically activate for the most probable observed action. Recalling that the presence of a goal (at least in working memory) is needed to elicit mirror neuron responses in the macaque. We believe it is also particularly important during the ontogenesis of the human mirror system, for example,

Woodward [62.122] has shown that even at nine months of age, infants recognized an action as being novel if it was directed toward a novel object rather than just having different kinematics, showing that the goal is more fundamental than the enacted trajectory. Similarly, if one sees someone drinking from a coffee mug then one can hypothesize that a particular action (that one already knows in motor terms) is used to obtain that particular effect. The association between the canonical response (object–action) and the mirror one (including vision) is made when the observed consequences (or goal) are recognized as similar in the two cases. Similarity can be evaluated following criteria ranging from kinematic to social consequences.

Many formulations of recognition tasks are available in the literature (see Duda, Hart, and Stork [62.123]) besides those keyed to the study of mirror neurons. Here, however, we focus on the Metta et al. [62.124] Bayesian interpretation of the recognition of actions. We equate the prior probabilities for actions with the object affordances, that is:

$$p(A_i|O_k), \quad (62.1)$$

where A_i is the i -th action from a motor repertoire of I actions and O_k is the target object of the grasping action out of a set of K possible objects. The affordances of an object identify the set of actions that are most likely to be executed upon it, and consequently the mirror activation of F5 can be thought as:

$$p(A_i|F, O_k), \quad (62.2)$$

where F are the features obtained by observation of the trajectory of the temporal evolution of the observed action. This probability can be computed from Bayes rule as

$$p(A_i|F, O_k) = p(F|A_i, O_k)p(A_i|O_k). \quad (62.3)$$

(An irrelevant normalization factor has been neglected so that, strictly speaking, the posterior in (62.3) is no longer a probability.) With this, a classifier is constructed by taking the maximum over the possible actions

$$\hat{A} = \max_i p(A_i|F, O_k). \quad (62.4)$$

Following Lopes and Santos-Victor [62.125], Metta et al. [62.124] assumed that the features F along the trajectories are independent. This is clearly not true for a smooth trajectory linking the movement of the hand and fingers to the observed action. However, this approximation simplified the estimation of the likelihoods

Table 62.1 Brain quantities and circuits

$p(A_i F, O_k)$	Mirror neuron responses, obtained by a combination of the information as in (62.3)
$p(F A_i, O_k)$	The activity of the F5 motor neurons generating certain motor patterns given the selected action and the target object
$p(A_i O_k)$	Object affordances: the response of the circuit linking AIP and the F5 canonical neurons $AIP \rightarrow F5$
Visuomotor map	Transformation of the hand-related visual information into motor data: identified with the response of $STS \rightarrow PF/PFG \rightarrow F5$, which is represented in F5 by $p(F A_i, O_k)$

in (62.2), though later implementations should take into account the dependence across time.

The object recognition stage (i.e., finding O_k) requires as much vision as is needed to determine the probability of the various grasp types being effective, the hand features F correspond to the STS response, and the response of the mirror neurons determines the most probable observed action A_i . We can identify certain circuits in the brain with the quantities described before.

However Table 62.1 does not exhaust the various computations required in the model. In the *learning*

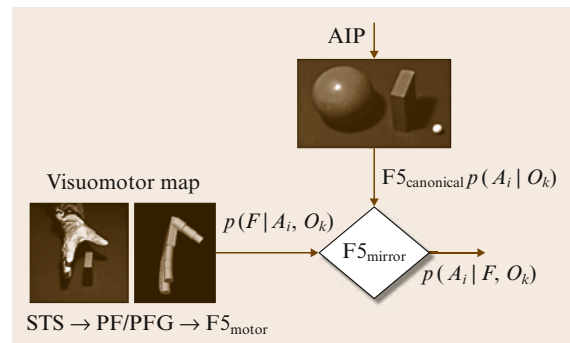


Fig. 62.8 Block diagram of the recognition process. Recognition (mirror neuron activation) is due to the convergence in F5 of two main contributions: signals from the AIP-F5 canonical connections and signals from the STS-PF/PFG-F5 circuit. In this model, activations are thought of as probabilities and are combined using Bayes's rule

phase, the visuomotor map is learned by a sigmoidal feedforward neural network trained with the backpropagation algorithm (both visual and motor signals are available); affordances are learned simply by counting the number of occurrences of actions given the object (visual processing of object features was assumed); and the likelihood was approximated by a mixture of Gaussians and the parameters learned by the expectation maximization (EM) algorithm. During the *recognition phase*, motor information is not available, but is recovered from the visuomotor map. Figure 62.8 shows a block diagram of the operation of the classifier.

A comparison (see Lopes and Santos-Victor [62.125]) was made of system performance: (a) when using the output of the inverse model and thus employing motor features to aid classification during the training phase, and (b) when only visual data were available for classification. Overall, their interpretation of the results is that by mapping in motor space they allow the classifier to choose features that are much better suited for performing optimally, which in turn facilitates general-

ization. The same is not true in visual space, since a given action may be viewed from different viewpoints. One may compare this to the viewpoint invariance of hand state adopted in the MNS model, which has the weakness there of being built in rather than emerging from training.

Another set of experiments was performed on a humanoid robot upper torso called Cog (see Brooks et al. [62.31]). Cog has a head, arms, and a moving waist for a total of 22 degrees of freedom but does not have hands. It has instead simple flippers that could be used to push and prod objects. Fitzpatrick and Metta [62.126] and Metta et al. [62.124] were interested in starting from minimal initial hypotheses yet yielding units with responses similar to mirror neurons. The robot was programmed to identify suitable cues to start the interaction with the environment and direct its attention towards potential objects, using an attention system similar to that (see Itti and Koch [62.56]) described in Sect. 62.2.5. Although the robot could not grasp objects, it could generate useful cues from touch, push, and prod actions.

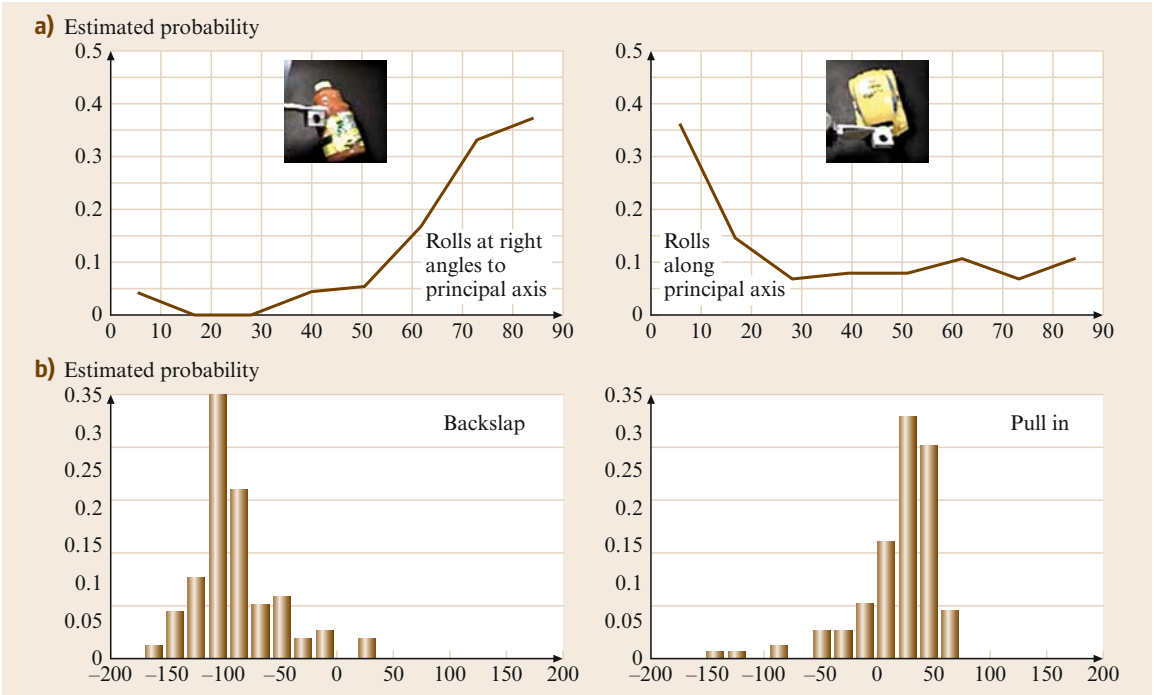


Fig. 62.9 (a) The affordances of two objects: a toy car and a bottle represented as the probability of moving along a given direction measured with respect to the object principal axis (visual). It is evident that the toy car tends to move mostly along the direction identified with the principal axis and the orange juice bottle at a right angle with respect to its principal axis. (b) The probability of a given action generating a certain object movement. The reference frame is anchored to the image plane in this case

The problem of defining objects reflects the problem of segmenting the visual scene. Criteria such as contrast, binocular disparity, and motion can be applied, but any of them can fail in a given situation. In the present system, optic flow was used to detect and segment an object after impact with the arm end point, and from segmentation based on shape, color, and behavior data. Depending on the available motor repertoire, the robot could explore a range of possible object behaviors (affordances) and form an object model which combines both sensorial and motor properties of the object that robot had the chance to interact with.

In a typical experiment, the human operator waves an object in front of the robot, which reacts by looking at it. If the object is dropped on the table in front of the robot, a reaching action is initiated, and the robot possibly makes contact with the object. Vision is used during the reaching and touching movement to guide the robot's flipper toward the object, to segment the hand from the object upon contact, and to collect information about the behavior of the object caused by the application of a certain action (see *Fitzpatrick* [62.127]). Unfortunately, interaction of the robot's flipper with objects does not result in a wide class of different affordances and so this study focused on the rolling affordances of a toy car, an orange juice bottle, a ball, and a colored toy cube. Besides reaching, the robot's motor repertoire consists of four different stereotyped approach movements covering a range of directions of about 180° around the object. For each successful trial, the robot *stored* the result of the segmentation, the object's principal axis which was selected as representative shape parameter, the action – initially selected randomly from the set of four approach directions – and the movement of the center of mass of the object for some hundreds of milliseconds after impact with the flipper was detected. This gives information about the rolling properties (affordances) of the different objects, e.g., the car tends to roll along its principal axis, the bottle at a right angle with respect to the axis. Figure 62.9 shows the result of collecting about 700 samples of generic poking actions and estimating the average direction of displacement of the object. Note, for example, that the action labeled as *backslap* (moving the object with the flipper outward from the robot) consistently gives a visual object motion upward in the image plane (corresponding to the peak at -100° , 0° being the direction parallel to the image x -axis; the y -axis pointing downward). A similar consideration applies to the other actions. Although crude, this implementation shows that with little pre-existing structure the robot could acquire the crucial elements for building knowledge of objects in

terms of their affordances. Given a sufficient level of abstraction, this implementation is close to the response of canonical neurons in F5 and their interaction with neurons observed in AIP that respond to object orientation (see *Sakata et al.* [62.128]).

62.4.3 Mirror Neurons and Imitation

Fitzpatrick and *Metta* [62.126] also addressed the question of what is further required for interpreting observed actions. Where in the previous section, the robot identified the motion of the object because of a specific action applied to it, here it could backtrack and derive the type of action from the observed motion of the object. It can further explore what is causing motion and learn about the concept of manipulator in a more general setting. In fact, the same segmentation procedure mentioned earlier could visually interpret poking actions generated by a human as well as those generated by the robot. One might argue that observation could be exploited for learning about object affordances. This is possibly true to the extent passive vision is reliable and action is not required. The advantage of the active approach, at least for the robot, is that it allows controlling the amount of information impinging on the visual sensors by, for instance, controlling the speed and type of action. This strategy might be especially useful given the limitations of artificial perceptual systems. Thus, observations can be converted into interpreted actions. The action whose effects are closest to the observed consequences on the object (which we might translate into the goal of the action) is selected as the most plausible interpretation given the observation. Most importantly, the interpretation reduces to the interpretation of the *simple* kinematics of the goal and consequences of the action rather than to understanding the *complex* kinematics of the human manipulator. The robot understands only to the extent it has learned to act. One might note that a more refined model should probably include visual cues from the appearance of the manipulator into the interpretation process. Indeed, the *hand state* that was central to the Oztot–Arbib model was based on an object-centered view of the hand's trajectory in a coordinate frame based on the object's affordances. The last question to address is whether the robot can imitate the *goal* of a poking action. The step is indeed small since most of the work is actually in interpreting observations. Imitation was generated in the following by replicating the latest observed human movement with respect to the object and irrespective of its orientation, for example, in case the experimenter poked the toy car sideways,

the robot imitated him/her by pushing the car sideways. Starting from this simple experiment, we need to formalize what is required for a system that has to acquire and deliver imitation.

Roboticians have been fascinated by the discovery of mirror neurons and the purported link to imitation that exists in the human nervous system. The literature on the topic extends from models of the monkey's (nonimitative) action recognition system (see *Oztop* and *Arbib* [62.112]) to models of the putative role of the mirror system in imitation (see *Demiris* and *Johnson* [62.119] and *Arbib* et al. [62.129]), and in real and virtual robots (see *Schaal* et al. [62.130]). *Oztop* et al. [62.131] propose a taxonomy of the models of the mirror system for recognition and imitation, and it is interesting to note how different the computational approaches that have been now framed as mirror system models are, including recurrent neural networks with parametric bias (see *Tani* et al. [62.132]), behavior-based modular networks (see *Demiris* and *Johnson* [62.119]), associative memory-based methods (see *Kuniyoshi* et al. [62.133]), and the use of multiple direct-inverse models as in the MOSAIC architecture (*Wolpert* et al. [62.134]; cf. the multiple paired forward-inverse models of Sect. 62.3.2).

Following the work of *Schaal* et al. [62.130] and *Oztop* et al. [62.131] we can propose a set of schemas required to produce imitation:

- determine what to imitate, inferring the goal of the demonstrator,
- establish a metric for imitation (correspondence; see *Nehaniv* [62.135]),
- map between dissimilar bodies (mapping),
- imitate behavior formation,

which are also discussed in greater detail by *Nehaniv* and *Dautenhahn* [62.136]. In practice, computational and robotic implementations have tackled these problems with different approaches and emphasizing different parts or specific subproblems of the whole, for example, in the work of *Demiris* and *Hayes* [62.137], the

rehearsal of the various actions (akin to the aforementioned theory of motor perception) was used to generate hypotheses to be compared with the actual sensory input. It is then remarkable how more recently a modified approach of this paradigm has been used in comparison with real human transcranial magnetic stimulation (TMS) data.

Ito et al. [62.138] (not the Masao Ito of cerebellar fame) took a dynamical systems approach using a recurrent neural network with parametric bias (RNNPB) to teach a humanoid robot to manipulate certain objects. In this approach the parametric bias (PB) encodes (tags) certain sensorimotor trajectories. Once learning is complete the neural network can be used either to recall a given trajectory by setting the PB externally or provide input for the sensory data only and observe the PB vector that would represent in that case the recognition of the situation on the basis of the sensory input only (no motor information available). It is relatively easy to interpret these two situations as the motor generation and the observation in a mirror neurons model.

The problem of building useful mappings between dissimilar bodies (consider a human imitating a bird's flapping wings) was tackled by *Nehaniv* and *Dautenhahn* [62.136] where an algebraic framework for imitation is described and the correspondence problem formally addressed. Any system implementing imitation should clearly provide a mapping between either dissimilar bodies or even in the case of similar bodies when either the kinematics or dynamics is different depending on the context of the imitative action.

Sausser and *Billard* [62.139] modeled the *ideomotor principle*, according to which observing the behavior of others influences our own performances. The ideomotor principle points directly to one of the core issues of the mirror system, that is, the fact that watching somebody else's actions changes something in the activation of the observer, thus facilitating certain neural pathways. The work in question also gives a model implemented in terms of neural fields (see *Sausser* and *Billard* [62.139] for details) and tries to explain the imitative cortical pathways and the behavior formation.

62.5 Extrodution

As the foregoing makes clear, robotics has much to learn from neuroscience and much to teach neuroscience. Neurorobotics can learn from the ways in which the brains and bodies of different creatures adapt to diverse ecological niches – as computa-

tional neuroethology helps us understand how the brain of a creature has evolved to serve *action-oriented perception*, and the attendant processes of learning, memory, planning, and social interaction.

We have sampled the *design* of just a few subsystems (both functional and structural) in just a few animals – optic flow in the bee, approach, escape, and barrier avoidance in frogs and toads, and navigation in the rat, as well as the control of eye movements in visual attention, the role of mammalian cerebellum in handling the nonlinearities and time delays of flexible motor systems, and the mirror systems of primates in action recognition and of humans in imitation. There are many more creatures with lessons to offer the roboticist than we can sample here.

Moreover, if we just confine attention to the brains of humans, this Chapter has mentioned at least 7a, 7b, AIP, area 46, caudo-putamen, cerebellum, cIPS, F2, F4, F5, hippocampus, hypothalamus, inferotemporal cortex,

LIP, MIP, motor cortex, nucleus accumbens, parietal cortex, prefrontal cortex, premotor cortex, pre-SMA (F6), spinal cord, STS, and VIP – and it is clear that there are many more details to be understood for each region, and many more regions whose interactions hold lessons for roboticists. We say this not to depress the reader, but rather to encourage further exploration of the literature of computational neuroscience and to note that the exchange with neurorobotics proceeds both ways: neuroscience can inspire novel robotic designs; conversely, robots can be used to test whether brain models still work when they make the transition from disembodied computer simulation to meeting the challenge of guiding the interactions of a physically embodied system with the complexities of its environment.

62.6 Further Reading

M.A. Arbib, (Ed.): *From Action to Language via the Mirror System* (Cambridge Univ. Press, Cambridge 2006).

This volume provides 16 articles on the mirror system, written by diverse experts. Of particular relevance to this Chapter are articles on dynamical systems: brain, body and imitation; attention and the minimal subscene; the development of grasping and the mirror system; and development of goal-directed imitation, object manipulation and language in humans and robots.

C. Bell, P. Cordo, S. Harnad: Controversies in neuroscience IV: motor learning and plasticity in the cerebellum. *Behavioral and brain sciences* **19**(3), (1996)

This somewhat older BBS special issue provides what was, back then, a rather definitive number of articles on the cerebellum, including an overview of models in a paper by Houk et al.

P. van der Smagt, D. Bullock: Applied intelligence, Scalable Applications of Neural Networks to Robotics **17**(1), (2002).

This special issue is focused on the application of cerebellar and other models to robotics tasks, and

lists some successful and – between the lines – more unsuccessful applications thereof.

V. Gallese, L. Fadiga, L. Fogassi, G. Rizzolatti: Action recognition in the premotor cortex. *Brain* **119**, 593–609 (1996)

This paper provides a detailed account of the neurophysiological evidence for mirror neurons. It is good reading to get the real data unbiased from further interpretation on the role of mirror neurons and it is complete and accurate. Although it is a technical paper it is easy to read also to a general audience.

L. Fadiga, L. Craighero, G. Buccino, G. Rizzolatti: Speech listening specifically modulates the excitability of tongue muscles: a TMS study. *Eur. J. Neurosci.* **15**(2), 399–402 (2002)

This work extends the mirror system concept with an interesting perspective on its role into language. This paper is interesting reading by providing evidence in humans (the other references above are about monkey experiments). In this case, it has been shown that speech listening facilitates the activation of tongue muscles which match the specific phoneme being listened to.

References

- 62.1 W.G. Walter: *The Living Brain* (Duckworth, London 1953), reprinted by Pelican Books, Harmondsworth, 1961
- 62.2 V. Braitenberg: *Vehicles: Experiments in Synthetic Psychology* (Bradford Books/The MIT Press, Cambridge 1984)

- 62.3 I. Segev, M. London: Dendritic Processing. In: *The Handbook of Brain Theory and Neural Networks*, ed. by M.A. Arbib (Bradford Books/The MIT Press, Cambridge 2003) pp. 324–332, Second Edition
- 62.4 Y. Fregnac: Hebbian synaptic plasticity. In: *The Handbook of Brain Theory and Neural Networks*, ed. by M.A. Arbib (Bradford Books/The MIT Press, Cambridge 2003) pp. 515–522, 2nd Edition
- 62.5 R.D. Beer: *Intelligence as Adaptive Behavior: An Experiment in Computational Neuroethology* (Academic, San Diego 1990)
- 62.6 D. Cliff: Neuroethology, computational. In: *The Handbook of Brain Theory and Neural Networks*, ed. by M.A. Arbib (Bradford Books/The MIT Press, Cambridge 2002), 2nd Edition
- 62.7 W. Reichardt: Autocorrelation, a principle for the evaluation of sensory information by the central nervous system. In: *Sensory Communication*, ed. by W.A. Rosenblith (MIT Press and Wiley, New York, London 1961) pp. 303–317
- 62.8 A. Borst, M. Dickinson: Visual course control in flies. In: *The Handbook of Brain Theory and Neural Networks*, ed. by M.A. Arbib (Bradford Books/The MIT Press, Cambridge 2003) pp. 1205–1210, 2nd edn.
- 62.9 P. van der Smagt, F. Groen: Visual feedback in motion. In: *Neural Systems for Robotics*, ed. by O. Omidvar, P. van der Smagt (Morgan Kaufmann, San Francisco 1997) pp. 37–73
- 62.10 P. van der Smagt: Teaching a robot to see how it moves. In: *Neural Network Perspectives on Cognition and Adaptive Robotics*, ed. by A. Browne (Institute of Physics Publishing, Bristol 1997) pp. 195–219
- 62.11 S.C. Liu, A. Usseglio-Viretta: Fly-like visuomotor responses of a robot using aVLSI motion-sensitive chips, *Biol. Cybern.* **85**(6), 449–457 (2001)
- 62.12 F. Ruffier, S. Viollet, S. Amic, N. Franceschini: Bio-inspired optical flow circuits for the visual guidance of micro air vehicles, *Inter. Symp. Circuits Syst. (ISCAS)* 2003, Vol. 3 (2003)
- 62.13 M.B. Reiser, M.H. Dickinson: A test bed for insect-inspired robotic control, *Philos. Trans. Math. Phys. Eng. Sci.* **361**(1811), 2267–2285 (2003)
- 62.14 M.V. Srinivasan, S. Zhang, J.S. Chahl: Landing strategies in honeybees, and possible applications to autonomous airborne vehicles, *Biol. Bull.* **200**(2), 216–221 (2001)
- 62.15 A. Barron, M.V. Srinivasan: Visual regulation of ground speed and headwind compensation in freely flying honey bees. *Apis mellifera* L, *J. Exp. Biol.* **209**(Pt 5), 978–984 (2006)
- 62.16 T. Vladusich, J.M. Hemmi, M.V. Srinivasan, J. Zeil: Interactions of visual odometry and landmark guidance during food search in honeybees, *J. Exp. Biol.* **208**, 4123–4135 (2005)
- 62.17 M.V. Srinivasan, S.W. Zhang: Visual control of honeybee flight, *EXS* **84**, 95–113 (1997)
- 62.18 B. Webb: Can robots make good models of biological behaviour?, *Behav. Brain Sci.* **24**, 1033–1094 (2001)
- 62.19 J.Y. Lettvin, H. Maturana, W.S. McCulloch, W.H. Pitts: What the frog's eye tells the frog brain, *Proc. IRE*, Vol. 47 (1959) pp. 1940–1951
- 62.20 D. Ingle: Visual releasers of prey catching behavior in frogs and toads, *Brain Behav. Evol.* **1**, 500–518 (1968)
- 62.21 R.L. Didday: A model of visuomotor mechanisms in the frog optic tectum, *Math. Biosci.* **30**, 169–180 (1976)
- 62.22 M.A. Arbib: Levels of modeling of visually guided behavior, *Behav. Brain Sci.* **10**, 407–465 (1987)
- 62.23 M.A. Arbib: Visuomotor coordination: neural models and perceptual robotics. In: *Visuomotor Coordination: Amphibians, Comparisons, Models, and Robots*, ed. by J.P. Ewert, M.A. Arbib (Plenum, New York 1989) pp. 121–171
- 62.24 T. Collett: Do toads plan routes? A study of detour behavior of *B. viridis*, *J. Compar. Physiol.* **146**, 261–271 (1982)
- 62.25 M.A. Arbib, D.H. House: Depth and detours: An essay on visually guided behavior. In: *Vision, Brain and Cooperative Computation* ed. by M.A. Arbib, A.R. Hanson (Bradford Books/The MIT Press, Cambridge 1987) pp. 129–163
- 62.26 F.J. Corbacho, M.A. Arbib: Learning to detour, *Adapt. Behav.* **4**, 419–468 (1995)
- 62.27 A. Cobas, M.A. Arbib: Prey-catching and predator-avoidance in frog and toad: defining the schemas, *J. Theor. Biol.* **157**, 271–304 (1992)
- 62.28 R.C. Arkin: Motor schema-based mobile robot navigation, *Int. J. Robot. Res.* **8**, 92–112 (1989)
- 62.29 O. Khatib: Real-time obstacle avoidance for manipulators and mobile robots, *Int. J. Robot. Res.* **5**, 90–98 (1986)
- 62.30 B.H. Krogh, C.E. Thorpe: Integrated path planning and dynamic steering control for autonomous vehicles, *Proc. IEEE Int. Conf. Robot. Autom.* (San Francisco 1986) pp. 1664–1669
- 62.31 R.A. Brooks, C.L. Breazeal, M. Marjanoviæ, B. Scassellati: The COG project: building a humanoid robot. In: *Computation for Metaphor, Analogy and Agents*, Vol. 1562, ed. by C.L. Nehaniv (Springer, New York 1999) pp. 52–87
- 62.32 R.C. Arkin: *Behavior-Based Robotics* (MIT Press, Cambridge 1998)
- 62.33 R.C. Arkin, M. Fujita, T. Takagi, R. Hasegawa: An ethological and emotional basis for human-Robot interaction, *Robot. Auton. Syst.* **42**(3–4), 191–201 (2003)
- 62.34 P. Dean, P. Redgrave, G.W.M. Westby: Event or emergency? Two response systems in the mammalian superior colliculus, *Trends Neurosci.* **12**, 138–147 (1989)
- 62.35 B.E. Stein, M.A. Meredith: *The Merging of the Senses* (MIT Press, Cambridge 1993)

- 62.36 T. Strosslin, C. Krebs, A. Arleo, W. Gerstner: Combining multimodal sensory input for spatial learning, *Artificial Neural Networks – ICANN 2002*, Lecture Notes Comput. Sci. **2415**, 87–92 (2002)
- 62.37 J. O’Keefe, L. Nadel: *The Hippocampus as a Cogn. Map* (Clarendon, Oxford 1978)
- 62.38 V. Braitenberg: Taxis, kinesis, decussation, *Progr. Brain Res.* **17**, 210–222 (1965)
- 62.39 A. Guazzelli, F.J. Corbacho, M. Bota, M.A. Arbib: Affordances, motivation, and the world graph theory, *Adapt. Behav.* **6**, 435–471 (1998)
- 62.40 J.J. Gibson: *The senses considered as perceptual systems* (Allen and Unwin, London 1966)
- 62.41 S.B. Choi, S.W. Ban, M. Lee: Biologically motivated visual attention system using bottom-up saliency map and top-down inhibition, *Neural Inf. Proc. – Lett. Rev.* **2**(1), 19–25 (2004)
- 62.42 M.A. Arbib: Perceptual structures and distributed motor control. In: *Handbook of Physiology – The Nervous System II. Motor Control*, ed. by V.B. Brooks (Am. Physiological Society, Bethesda 1981) pp.1449–1480.
- 62.43 A. Guazzelli, F.J. Corbacho, M. Bota, M.A. Arbib: Affordance, motivation, and the world graph theory, *Adapt. Behav.* **6**, 435–471 (1998)
- 62.44 I. Lieblisch, M.A. Arbib: Multiple representations of space underlying behavior, *Behav. Brain Sci.* **5**, 627–659 (1982)
- 62.45 B. Girard, D. Filliat, J.A. Meyer, A. Berthoz, A. Guillot: Integration of navigation and action selection functionalities in a computational model of cortico-basal-ganglia-thalamo-cortical loops, *Adapt. Behav.* **13**(2), 115–130 (2005)
- 62.46 J.A. Meyer, A. Guillot, B. Girard, M. Khamassi, P. Pirim, A. Berthoz: The *Psikharpax* project: towards building an artificial rat, *Robot. Auton. Syst.* **50**(4), 211–223 (2005)
- 62.47 D.M. Lyons, M.A. Arbib: A formal model of computation for sensory-based robotics, *IEEE Trans. Robot. Autom.* **5**, 280–293 (1989)
- 62.48 G. Metta, P. Fitzpatrick, L. Natale: YARP: yet another robot platform, *Int. J. Adv. Robot. Syst.* **3**(1), 43–48 (2006)
- 62.49 L.G. Ungerleider, M. Mishkin: Two cortical visual systems. In: *Analysis of Visual Behavior*, ed. by D.J. Ingle, M.A. Goodale, R.J.W. Mansfield (MIT Press, Cambridge 1982) pp. 549–586
- 62.50 M.A. Goodale, A.D. Milner: Separate visual pathways for perception and action, *Trends Neurosci.* **15**, 20–25 (1992)
- 62.51 M. Jeannerod, B. Biguer: Visuomotor mechanisms in reaching within extra-personal space, In: *Advances in the Analysis of Visual Behavior*, ed. by D.J. Ingle, R.J.W. Mansfield, M.A. Goodale (MIT Press, 1982) pp. 387–409 pp. 285–306
- 62.52 B. Hoff, M.A. Arbib: Simulation of interaction of hand transport and preshape during visually guided reaching to perturbed targets, *J. Motor Behav.* **25**, 175–192 (1993)
- 62.53 B. Hoff, M.A. Arbib: A model of the effects of speed, accuracy, and perturbation on visually guided reaching. In: *Control of Arm Movement in Space: Neurophysiological and Computational Approaches*, Experimental Brain Research Series, Vol. 22, ed. by R. Caminiti, P.B. Johnson, Y. Burnod (Springer, New York 1992)
- 62.54 R.C. Miall, D.J. Weir, D.M. Wolpert, J.F. Stein: Is the cerebellum a Smith predictor?, *J. Motor Behav.* **25**, 203–216 (1993)
- 62.55 G. Deco, E.T. Rolls: Attention and working memory: a dynamical model of neuronal activity in the prefrontal cortex, *Eur. J. Neurosci.* **18**(8), 2374–2390 (2003)
- 62.56 L. Itti, C. Koch: A saliency-based search mechanism for overt and covert shifts of visual attention, *Vision Res.* **40**, 1489–1506 (2000)
- 62.57 J.M. Wolfe: Guided search 2.0: a revised model of visual search, *Psychonomic Bull. Rev.* **1**, 202–238 (1994)
- 62.58 A. Yarbus: *Eye Movements and Vision* (Plenum, New York 1967)
- 62.59 V. Navalpakkam, L. Itti: Modeling the influence of task on attention, *Vision Res.* **45**, 205–231 (2005)
- 62.60 F. Orabona, G. Metta, G. Sandini: Object-based visual attention: a model for a behaving robot, *Conf. Comput. Vision Pattern Recognition* (2005) pp. 89–89
- 62.61 G. Sandini, V. Tagliasco: An anthropomorphic retina-like structure for scene analysis, *Comput. Vision, Graphics Image Proc.* **14**(3), 365–372 (1980)
- 62.62 G. Rizzolatti, L. Riggio, I. Dascola, C. Umiltà: Reorienting attention across the horizontal and vertical meridians: evidence in favor of a premotor theory of attention, *Neuropsychologia* **25**, 31–40 (1987)
- 62.63 J.R. Flanagan, R.S. Johansson: Action plans used in action observation, *Nature* **424**, 769–771 (2003)
- 62.64 J.R. Flanagan, P. Vetter, R.S. Johansson, D.M. Wolpert: Prediction precedes control in motor learning, *Curr. Biol.* **13**, 146–150 (2003)
- 62.65 J.M. Mataric, M. Pomplun: Fixation behavior in observation and imitation of human movement, *Brain Res. Cogn. Brain Res.* **7**, 191–202 (1998)
- 62.66 J.C. Eccles, M. Ito, J. Szentágothai: *The Cerebellum as a Neuronal Machine* (Springer, New York 1967)
- 62.67 M. Ito: Cerebellar circuitry as a neuronal machine, *Progr. Neurobiol.* **78**(4), 272–303 (2006)
- 62.68 D.A. Marr: A theory of cerebellar cortex, *J. Physiol.* **202**, 437–470 (1969)
- 62.69 J.S. Albus: A theory of cerebellar function, *Math. Biosci.* **10**, 25–61 (1971)
- 62.70 J.S. Albus: Data storage in the cerebellar model articulation controller (CMAC), *J. Dyn. Syst. Meas. Contr. ASME* **3**, 228–233 (1975)
- 62.71 W.T. Miller: Real-time application of neural networks for sensor-based control of robots with

- vision, IEEE Trans. Syst. Man Cybern. **19**, 825–831 (1994)
- 62.72 J. Peters, P. van der Smagt: Searching a scalable approach to cerebellar based control, Appl. Intell. **17**, 11–33 (2002)
- 62.73 E.J. Nijhof, E. Kouwenhoven: Simulation of multi-joint arm movements. In: *Biomechanics and Neural Control of Posture and Movement*, ed. by J.M. Winters, P.E. Crago (Springer, New York 2002) pp. 363–372
- 62.74 M. Damsgaard, J. Rasmussen, S.T. Christensen: Numerical simulation and justification of antagonists in isometric squatting, Proceedings of the 12th Conference of the European Society of Biomechanics, ed. by P.J. Prendergast, T.C. Lee, A.J. Carr (Royal Academy of Medicine in Ireland, 2000)
- 62.75 D. Bullock, J. Contreras-Vidal: How spinal neural networks reduce discrepancies between motor intention and motor realization. In: *Variability and Motor Control*, ed. by K. Newell, D. Corcos (Human Kinetics, Champaign 1993) pp. 183–221
- 62.76 S. Schaal, N. Schweighofer: Computational motor control in humans and robots, Curr. Opin. Neurobiol. **15**, 675–682 (2005)
- 62.77 P. van der Smagt, G. Hirzinger: The cerebellum as computed torque model, Fourth Int. Conf. Knowledge-Based Intell. Eng. Syst. Allied Technol., Vol. 2 (2000) pp. 760–763
- 62.78 M. Ebadzadeh, B. Tondou, C. Darlot: Computation of inverse functions in a model of cerebellar and reflex pathways allow to control a mobile mechanical segment, Neuroscience **133**, 29–49 (2005)
- 62.79 M. Ito: *The Cerebellum and Neural Control* (Raven, New York 1984)
- 62.80 P. van der Smagt: Cerebellar control of robot arms, Connect. Sci. **10**, 301–320 (1998)
- 62.81 P. van der Smagt: Benchmarking cerebellar control, Robot. Auton. Syst. **32**, 237–251 (2000)
- 62.82 C. Sabourin, O. Bruneau: Robustness of the dynamic walk of a biped robot subjected to disturbing external forces by using CMAC neural networks, Robot. Auton. Syst. **51**, 81–99 (2005)
- 62.83 J.C. Houk, J.T. Buckingham, A.G. Barto: Models of the cerebellum and motor learning, Behav. Brain Sci. **19**(3), 368–383 (1996)
- 62.84 M.A. Arbib, C.C. Boylles, P. Dev: Neural models of spatial perception and the control of movement. In: *Kybernetik und Bionik/Cybernetics* (Oldenbourg, 1974) pp. 216–231
- 62.85 N. Schweighofer: *Computational Models of the Cerebellum in the Adaptive Control of Movements* (University of Southern California, Los Angeles 1995), Ph.D. thesis
- 62.86 N. Schweighofer, M.A. Arbib, M. Kawato: Role of the cerebellum in reaching quickly and accurately: I. A functional anatomical model of dynamics control, Eur. J. Neurosci. **10**, 86–94 (1998)
- 62.87 N. Schweighofer, J. Spoelstra, M.A. Arbib, M. Kawato: Role of the cerebellum in reaching quickly and accurately: II. A neural model of the intermediate cerebellum, Eur. J. Neurosci. **10**, 95–105 (1998)
- 62.88 M.A. Arbib, N. Schweighofer, W.T. Thach: Modeling the cerebellum: from adaptation to coordination. In: *Motor Control and Sensory-Motor Integration: Issues and Directions*, ed. by D.J. Glencross, J.P. Piek (North-Holland Elsevier Science, Amsterdam 1995) pp. 11–36
- 62.89 D. Wolpert, M. Kawato: Multiple paired forward and inverse models for motor control, Neural Networks **11**, 1317–1329 (1998)
- 62.90 P. van der Smagt, F. Groen, K. Schulten: Analysis and control of a rubber-tuator robot arm, Biol. Cybern. **75**(4), 433–440 (1996)
- 62.91 R. Osu, H. Gomi: Multijoint muscle regulation mechanisms examined by measured human arm stiffness and EMG signals, J. Neurophysiol. **81**(4), 1458–1468 (1999)
- 62.92 G. Rizzolatti, L. Fadiga, V. Gallese, L. Fogassi: Premotor cortex and the recognition of motor actions, Cogn. Brain Res. **3**, 131–141 (1995)
- 62.93 V. Gallese, L. Fadiga, L. Fogassi, G. Rizzolatti: Action recognition in the premotor cortex, Brain **119**, 593–609 (1996)
- 62.94 S.T. Grafton, M.A. Arbib, L. Fadiga, G. Rizzolatti: Localization of grasp representations in humans by PET: 2. Observation compared with imagination, Exp. Brain Res. **112**, 103–111 (1996)
- 62.95 G. Rizzolatti, L. Fadiga, M. Matelli, V. Bettinardi, D. Perani, F. Fazio: Localization of grasp representations in humans by positron emission tomography: 1. Observation versus execution, Exp. Brain Res. **111**, 246–252 (1996)
- 62.96 L. Fadiga, G. Buccino, L. Craighero, L. Fogassi, V. Gallese, G. Pavesi: Corticospinal excitability is specifically modulated by motor imagery: a magnetic stimulation study, Neuropsychologia **37**, 147–158 (1999)
- 62.97 B. Voelkl, L. Huber: Imitation as faithful copying of a novel technique in marmoset monkeys, PLoS ONE **2**(7), e611 (2007)
- 62.98 R.W. Byrne: Imitation as behavior parsing, Philos. Trans. R. Soc. London **358**, 529–536 (2003)
- 62.99 M. Iacoboni, R.P. Woods, M. Brass, H. Bekkering, J.C. Mazziotta, G. Rizzolatti: Cortical mechanisms of human imitation, Science **286**, 2526–2528 (1999)
- 62.100 M.A. Arbib, G. Rizzolatti: Neural expectations: a possible evolutionary path from manual skills to language, Commun. Cognition **29**, 393–424 (1997)
- 62.101 M.A. Arbib: From monkey-like action recognition to human language: an evolutionary framework for neurolinguistics (with commentaries and author's response), Behav. Brain Sci. **28**, 105–167 (2005)

- 62.102 A.M. Liberman, F.S. Cooper, D.P. Shankweiler, M. Studdert-Kennedy: Perception of the speech code, *Psychol. Rev.* **74**, 431–461 (1967)
- 62.103 G. Rizzolatti, G. Luppino, M. Matelli: The organization of the cortical motor system: new concepts, *Electroencephalogr. Clin. Neurophysiol.* **106**, 283–296 (1998)
- 62.104 G. Rizzolatti, L. Fogassi, V. Gallese: Neurophysiological mechanisms underlying the understanding and imitation of action, *Nat. Rev. Neurosci.* **2**, 661–670 (2001)
- 62.105 M.A. Umiltà, E. Kohler, V. Gallese, L. Fogassi, L. Fadiga, C. Keysers, G. Rizzolatti: I know what you are doing. A neurophysiological study, *Neuron* **31**(1), 155–165 (2001)
- 62.106 P.F. Ferrari, S. Rozzi, L. Fogassi: Mirror neurons responding to observation of actions made with tools in monkey ventral premotor cortex, *J. Cogn. Neurosci.* **17**, 212–226 (2005)
- 62.107 L. Fogassi, P.F. Ferrari, B. Gesierich, S. Rozzi, F. Chersi, G. Rizzolatti: Parietal Lobe: From action organization to intention understanding, *Science* **308**(4), 662–667 (2005)
- 62.108 M. Taira, S. Mine, A.P. Georgopoulos, A. Murata, H. Sakata: Parietal cortex neurons of the monkey related to the visual guidance of hand movement, *Exp. Brain Res.* **83**, 29–36 (1990)
- 62.109 A.H. Fagg, M.A. Arbib: Modeling parietal-premotor interactions in primate control of grasping, *Neural Netw.* **11**, 1277–1303 (1998)
- 62.110 G. Rizzolatti, G. Luppino: Grasping movements: visuomotor transformations. In: *The Handbook of Brain Theory and Neural Networks*, ed. by M.A. Arbib (MIT Press, Cambridge 2003) pp. 501–504, 2nd ed
- 62.111 S.H. Johnson-Frey: The neural bases of complex tool use in humans, *Trends Cogn. Sci.* **8**, 71–78 (2004)
- 62.112 E. Oztop, M.A. Arbib: Schema design and implementation of the grasp-related mirror neuron system, *Biol. Cybern.* **87**(2), 116–140 (2002)
- 62.113 D.I. Perrett, J.K. Hietanen, M.W. Oram, P.J. Benson: Organization and functions of cells in the macaque temporal cortex, *Philos. Trans. Royal Society London, B* **335**, 23–50 (1992)
- 62.114 D.P. Carey, D.I. Perrett, M.W. Oram: Recognizing, understanding and producing action. In: *Handbook of Neuropsychology: Action and Cognition*, Vol. 11, ed. by M. Jeannerod, J. Grafman (Elsevier, Amsterdam 1997) pp. 111–130
- 62.115 B. Bonaiuto, E. Rosta, M.A. Arbib: Extending the mirror neuron system model, I. Audible actions and invisible grasps, *Biol. Cybern.* **96**(1), 9–38 (2006)
- 62.116 D.M. Wolpert, Z. Ghahramani, R.J. Flanagan: Perspectives and problems in motor learning, *Cogn. Sci.* **5**(11), 487–494 (2001)
- 62.117 M.I. Jordan, D.E. Rumelhart: Forward models: supervised learning with a distal teacher, *Cogn. Sci.* **16**(3), 307–354 (2006)
- 62.118 M. Kawato, K. Furukawa, R. Suzuki: A hierarchical neural network model for control and learning of voluntary movement, *Biol. Cybern.* **57**, 169–185 (1987)
- 62.119 Y. Demiris, M.H. Johnson: Distributed, predictive perception of actions: a biologically inspired robotics architecture for imitation and learning, *Connect. Sci.* **15**(4), 231–243 (2003)
- 62.120 M. Haruno, D.M. Wolpert, M. Kawato: MOSAIC model for sensorimotor learning and control, *Neural Comput.* **13**, 2201–2220 (2001)
- 62.121 E. Oztop, D.M. Wolpert, M. Kawato: Mental state inference using visual control parameters, *Cogn. Brain Res.* **22**, 129–151 (2005)
- 62.122 A.L. Woodward: Infant selectively encode the goal object of an actor's reach, *Cognition* **69**, 1–34 (1998)
- 62.123 R.O. Duda, P.E. Hart, D.G. Stork: *Pattern Classification*, 2nd edn. (Wiley, New York 2001)
- 62.124 G. Metta, G. Sandini, L. Natale, L. Craighero, L. Fadiga: Understanding mirror neurons: a bio-robotic approach, *Interaction Studies* **7**(2), 197–232 (2006), special issue on Epigenetic Robotics
- 62.125 M. Lopes, J. Santos-Victor: Visual learning by imitation with motor representations, *IEEE Trans. Syst. Man Cybern., Part B Cybern.* **35**(3), 438–449 (2005)
- 62.126 P. Fitzpatrick, G. Metta: Grounding vision through experimental manipulation, *Philos. Trans. R. Soc. Math. Phys. Eng. Sci.* **361**(1811), 2165–2185 (2003)
- 62.127 P. Fitzpatrick: First contact: an active vision approach to segmentation, *IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)* (Las Vegas 2003)
- 62.128 H. Sakata, M. Taira, M. Kusunoki, A. Murata, Y. Tanaka: The TINS lecture – The parietal association cortex in depth perception and visual control of action, *Trends Neurosci.* **20**(8), 350–358 (1997)
- 62.129 M.A. Arbib, A. Billard, M. Iacoboni, E. Oztop: Synthetic brain imaging: grasping, mirror neurons and imitation, *Neural Netw.* **13**(8–9), 975–997 (2000)
- 62.130 S. Schaal, A.J. Ijspeert, A. Billard: Computational approaches to motor learning by imitation, *Philos. Trans. R. Soc. Biol. Sci.* **358**(1431), 537–547 (2003)
- 62.131 E. Oztop, M. Kawato, M.A. Arbib: Mirror neurons and imitation: a computationally guided review, *Neural Netw.* **19**(3), 254–271 (2006)
- 62.132 J. Tani, M. Ito, Y. Sugita: Self-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robot experiments using RNNPB, *Neural Netw.* **17**(8–9), 1273–1289 (2004)
- 62.133 Y. Kuniyoshi, Y. Yorozu, M. Inaba, H. Inoue: From visuomotor self learning to visual imitation – a neural architecture for humanoid learning, *IEEE Int. Conf. Robot. Autom. (ICRA)*, Vol. 3 (2003) pp. 3132–3139

- 62.134 D.M. Wolpert, K. Doya, M. Kawato: A unifying computational framework for motor control and social interaction, *Phil. Trans. R. Soc. Biol. Sci.* **358**(1431), 593–602 (2003)
- 62.135 C.L. Nehaniv: Nine billion correspondence problems. In: *Imitation and Social Learning in Robots, Humans, and Animals: Behavioural, Social and Communicative Dimensions*, ed. by C.L. Nehaniv, K. Dautenhahn (Cambridge Univ. Press, Cambridge 2006)
- 62.136 C.L. Nehaniv, K. Dautenhahn: Mapping between dissimilar bodies: affordances and the algebraic foundations of imitation, *Eur. Workshop Learning Robots (EWRL-7)* (Edinburgh 1998)
- 62.137 Y. Demiris, G. Hayes: Imitation as a dual-route process featuring predictive and learning components: a biologically-plausible computational model. In: *Imitation in Animals and Artifacts*, ed. by K. Dautenhahn, C. Nehaniv (MIT Press, Cambridge 2002)
- 62.138 M. Ito, K. Noda, Y. Hoshino, J. Tani: Dynamic and interactive generation of object handling behaviors by a small humanoid robot using a dynamic neural network model,
- 62.139 E.L. Sauser, A. Billard: Parallel and distributed neural models of the ideomotor principle: An investigation of imitative cortical pathways, *Neural Netw.* **19**, 285–298 (2006)

Perceptual Robotics

Heinrich H. Bülthoff, Christian Wallraven, Martin A. Giese

Perceptual functions are central to many applications in robotics and for the construction of efficient human–robot interfaces. The study of perception in biological systems has revealed important information–processing principles that have been converted to powerful applications in robotics and computer vision. The chapter first discusses two central theories of object recognition: model- and exemplar-based theories. A review of experimental results from the study of object recognition in biological systems suggests that exemplar-based approaches capture important properties of object recognition in the brain. We then discuss how very similar principles have been realized in highly efficient technical systems for object recognition and detection, including realizations that are based on biologically inspired neural architectures. Principles for the efficient processing of complex shapes can be extended to the representation of complex movements and actions. We illustrate this by first reviewing some properties of the cortical mechanisms of the recognition of complex movements and actions, focusing on principles that are useful for robotics applications. Again, exemplar-based approaches seem to capture important properties of motion recognition in the brain, and at the same time provide a powerful approach for building technical movement recognition systems. Finally, it is shown that the example-based framework is not only useful for recognition, but also provides the basis for powerful synthesis methods. As one example we discuss the synthesis of photorealistic three-dimensional (3-D) models of faces, exploiting correspondence

63.1 Overview.....	1481
63.2 Example-Based Object Representations ..	1483
63.2.1 Perceptual and Computational Basis of Object Representations	1483
63.2.2 Neural Representations in Object Recognition	1485
63.2.3 Object Recognition: Lessons from Computer Vision.....	1486
63.2.4 Object Learning and Recognition for Perceptual Robotics	1488
63.3 Example-Based Movement Representations	1490
63.3.1 Recognition of Complex Movements in Visual Cortex.....	1490
63.3.2 Example-Based Motion Recognition in Robots and Computer Vision.....	1491
63.4 Example-Based Synthesis Models: From Faces to Movements	1492
63.4.1 Face Synthesis: Realistic Faces for Avatars and Robots.....	1492
63.4.2 Example-Based Trajectory Synthesis	1493
63.5 Conclusions and Further Reading.....	1494
References	1495

between training examples. Related approaches have been developed for spatiotemporal patterns. We review a class of algorithms that permit the accurate modeling of movements and movement styles by interpolation between example trajectories with high relevance for the synthesis of movements, e.g., in humanoid robotics.

63.1 Overview

The technical realization of perceptual functions is a central problem for many applications in robotics. Robots

require perception to navigate in space and to localize and recognize goal objects, e.g., for manipulation (see,

e.g., Chap. 15 and Part D). Social interactive robots must be able to interpret gestures, actions, and even emotions (Chap. 58) in order to interact naturally with their users. One important approach for the programming of complex perceptual and behavioral functions such as those needed by humanoid robots is imitation learning (Chaps. 59 and 62). Imitation learning requires the robot to *perceive* complex actions that are executed by the user and to subsequently map them into an efficient representation that is suitable for the synthesis of the corresponding motor behavior on the available platform. This chapter focuses on important principles of the representation of complex shapes and movements, which can be derived from biological perception systems, and more specifically the basic functionality of the primate visual cortex. Such principles have interesting implications for the design of technical systems in robotics and computer vision for the *recognition of objects, shapes, and faces*, and for the recognition and synthesis of *complex movements and actions*. The limited space of the chapter forces us to focus mainly on visual perception and related technical applications. In the context of robotics many other aspects of perception are important, for example haptic perception (Chap. 30), auditory perception, sensory cue fusion (Chap. 25), or the interaction between the visual recognition of objects and actions and motor programs, e.g., during grasping (specifically treated in Chap. 62).

In the following, we will first formulate several biological principles that are relevant for form and motion representations, specifically in the visual system. We will then, on the one hand, describe technical systems that implement these principle using neural mechanisms that are inspired by the basic architecture of the brain. On the other hand, we will also discuss implementations that are inspired by biological principles on a more abstract level, and which exploit instead of neural networks more efficient technical algorithms for the realization of biologically relevant functions. Many of these systems were derived in the field of computer vision and are based on the advantages and limitations of modern digital computers in order to realize biological principles of information processing more efficiently.

Our approach to establish relationships between biological perception and robotics systems at different levels reflects *David Marr's* classical distinction of multiple levels of description, originally developed for the analysis of vision systems [63.1]: robotics systems can be inspired by biological system at the level of *implementation*, i.e., one can try to build robots containing

neural mechanisms that imitate the function of neurons in the central nervous systems of biological organisms. This type of analogy between technical and biological systems coincides with the definition of ‘neurorobotics’ given in Chap. 62. A transfer of principles from biological perception systems to robots might also be accomplished at the more abstract levels of *computational problems* and *algorithms*. The computational level is defined by the abstract theoretical formulation of computational problems that have to be solved by perception systems. Examples are the identification or classification of goal object, or the recognition of human gestures. Marr’s level of algorithms specifies the computational methods for the solution of such problems, independent of the underlying specific hardware or architecture. For example, an object might be represented by modeling its full 3-D structure, e.g., using a parametric 3-D shape model, or it might be represented in terms of two-dimensional (2-D) example views. Example views, however, might be represented using neural networks, establishing an analogy with the human brain at the level of implementation, or using more efficient computational methods, e.g., as support vectors of a classifier that has been trained with appropriate images of the object and distractor patterns. In both cases, the robot system realizes mechanisms that are derived from perception in biological systems.

Marr’s distinction of levels is only one way to introduce description levels for complex systems. Other approaches, particularly relevant for robotics, are for example the subsumption architecture [63.2] and behavior-based approaches (Chap. 38) that decompose robotics system into a system of simpler behavioral modules. Another example are dynamical systems approaches to robotics [63.3–5] that are based on the biologically motivated idea that behaviors can be mapped onto stable states of (nonlinear) dynamical systems or recurrent neural networks. Individual behaviors result from self-organization over the whole system as collectively stable modes, which can be described and analyzed by the introduction of appropriate collective variables. Interestingly, such robotics-inspired approaches have been quite successful in modeling human navigation behavior [63.6].

In the following we will apply the term *perceptual robotics* to signify the design of robots based on principles that are derived from human perception on all three levels in the sense of Marr. This includes a realization in terms of specific neural circuits as well as the transfer

of more abstract biologically inspired strategies for the solution of relevant computational problems. A direct interaction between robotics and perception research can be very fruitful for both disciplines. On the one hand, our current knowledge about human perception and the underlying computational principles might help to build more efficient robotics architectures that inherit properties from biological perception, e.g., very efficient and robust processing or complex dynamic flexibility. Such architectures will be a necessary prerequisite for the creation of truly intelligent, cognitive robots (Chaps. 9, 38, and 58). On the other hand, perception science

often uses robots as a test bed for gaining a deeper understanding of computational processes, and in particular for testing the computational power of specific computational solutions under *real-world conditions*. How can a child, for example, learn how to handle new objects, and what allows us to learn the visual categorization of thousands of objects from just a few examples? *Perceptual robot platforms*, equipped with a variety of sensory inputs and operating in different types of artificially structured or real-world environments, provide very helpful tools for the study of such questions.

63.2 Example-Based Object Representations

Object recognition is a fundamental visual function that is critical for many applications in robotics. Manipulation and grasping (Chaps. 26 and 28) require exact knowledge about the shape of the goal object that is often derived from visual sensors. Also the imitation of goal directed movements (Chap. 59) requires knowledge about target objects. Finally, social and collective robots require robust recognition of other agents and objects which are taking part in the present action (Chaps. 41 and 58). The importance of object and shape recognition for many other applied robot systems, like construction and assembly robots or smart cars (Chap. 51) is immediately evident.

63.2.1 Perceptual and Computational Basis of Object Representations

The question of how humans learn, represent, and recognize objects under a wide variety of viewing conditions presents a great challenge to both neurophysiology and cognitive research. Frameworks for explaining the amazing robustness of human recognition processes and how humans represent objects can be broadly classified into two approaches: in the *model-based representation*, an image on the retina is analyzed to yield three-dimensional parts of an object based on geometric primitives (Chap. 23). These primitives are then matched to an internal, three-dimensional model of the object (Fig. 63.1). *Exemplar-based representation* approaches assume that the internal storage consists of, typically two-dimensional, snapshot-like representations of objects, which are directly compared to the visual input via simple image transformations. In the following, we

will briefly describe the basic properties of these two approaches as well as perceptual evidence for their plausibility in explaining human recognition performance.

Structural Description Models

The basic idea of structural description models is that object recognition or categorization is based on a structural representation, which is defined as a configuration of elementary object parts that are regarded as *shape primitives* [63.7]. Structural description models aim to supply abstract and propositional descriptions of objects, while at the same time disregarding irrelevant spatial information. Therefore, structural description models typically predict that recognition performance is invariant regarding spatial transformations. Biederman's recognition-by-components (RBC) or geon structural description (GSD) model can be regarded as the best developed example of the structural description model type [63.8]. According to this model objects are represented as configurations of elementary three-dimensional primitive parts, called geons. These geons are derived from nonaccidental properties (NAPs) in the image, i.e., from properties which unlikely arise by chance, and are more or less invariant over a wide range of views. For example, the properties straight versus curved, symmetrical versus asymmetrical, parallel versus nonparallel are regarded as nonaccidental properties (nonaccidental properties were originally proposed within an image-based approach by Lowe [63.9]). According to the model, geons and their spatial configuration are combined into a structural representation, called geon structural description. The spatial relations between parts are described in a categorical way, using relations such as *above*, *below*, etc. Like other struc-

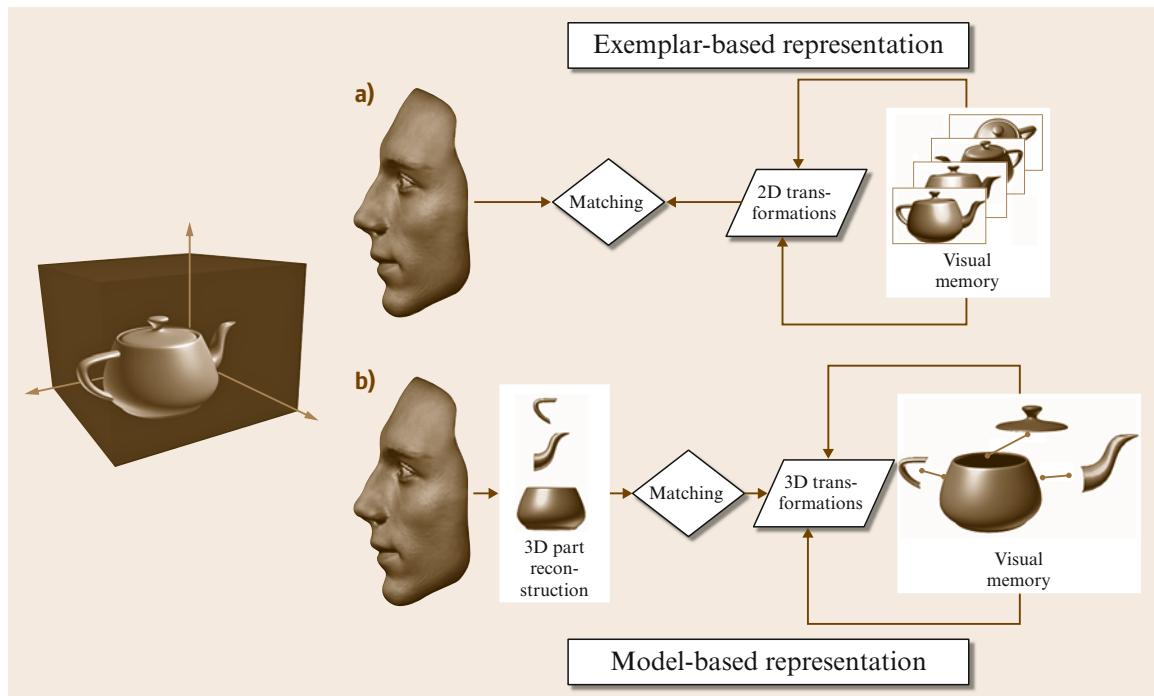


Fig. 63.1a,b Schematic drawing comparing exemplar-based (a) and model-based (b) representations. Object perception based on model-based representations assumes that the brain extracts 3-D parts from the visual image, which are then matched to an internally stored 3-D model of the teapot. Contrasting with this approach, object perception based on exemplar-based representations is accomplished by directly comparing stored templates or example images with the current picture of the teapot

tural description models, Biederman's model predicts invariance in relation to position and size and also in relation to orientation in depth, as long as no parts are occluded.

The question has to be raised whether objects can be decomposed into geons at all. It was argued that Biederman's RBC cannot be applied to a whole range of biological stimuli [63.10], or that biological shapes in general cannot be adequately described by structural description models [63.11]. This problem extends also to artefact categories such as *shoe*, *hat*, or *backpack*, which seem to exceed the scope of the geon model. Therefore it has to be doubted that object parts are necessarily represented as geons, or as similar geometrical primitives (for further problems of RBC see [63.12, 13]). However, this does not mean that category representations do not have a part structure: in fact it is not the notion of part structure in object representations by itself which is problematic, but the use of parts and relations as a basis to derive invariant recognition performance [63.14].

Exemplar-Based Models

Over the last two decades, an increasing number of studies has demonstrated that recognition is not view-independent. Orientation-dependent recognition effects were found for novel objects [63.15, 16], and also for common, familiar objects [63.17, 18]. Orientation-dependent recognition performance has been shown not to be limited to individual objects, such as faces [63.19, 20], or to objects on the subordinate level of categorization [63.15, 21], but was also demonstrated for basic-level recognition [63.18, 22].

Moreover, recognition performance is not only influenced by orientation, but also by the *size* of the stimulus. Results are quite similar: reaction times (RTs) and error rates depend on the extent of transformation that is necessary to align memory and stimulus representation. RTs increase in a monotonic way with increasing change of (perceived) size (for a review see [63.23]). Several studies even show a systematic relationship between the amount of *translation* and recognition performance:

Increasing displacement between two sequentially presented stimuli led to a deterioration of performance, both for novel objects [63.24] and familiar objects [63.25]. Overall, view-independent models are difficult to reconcile with these findings which indicate that recognition performance depends systematically on different spatial transformations.

In the following, we will briefly review three types of exemplar-based models, which – by virtue of different computational mechanisms and processes (including alignment, interpolation, and pooling/thresholding) – explain the transformation-dependent performance that was found in the psychophysical experiments.

In the class of *alignment models*, Ullman's [63.10, 26] 3-D alignment model and Lowe's [63.9] SCERPO model are probably the best known examples. Both models work by storing 3-D models of objects, which are aligned to images by perspective projection of corresponding features (edges or feature points on the object). As an alternative to Ullman's [63.26] model, which relies on 3-D object representations, Ullman and Basri [63.27] suggested an alignment model on the basis of 2-D views. In this model, an internal object model is constructed by a linear combination of a small number of stored 2-D exemplar images. Thus, the alignment is not achieved by a spatial compensation process, but by a linear combination of images. The intuition behind the linear combination approach can be explained in simple terms: suppose that two views of the same three-dimensional object are stored, taken from somewhat different viewing directions. An intermediate view can then be described as a weighted sum of the views that are already stored. In this case, the representation is based on the two-dimensional positions of corresponding features in each view. Making the set of views closer results in an object representation that is equivalent to storing a 3-D model.

In the *interpolation model*, recognition is achieved by localization in a multidimensional representational space, which is spanned by stored views [63.28]. The interpolation model is based on the theory of approximation of multivariate functions and can be implemented with radial basis functions (RBFs). In this scheme, the whole viewing space of an object is approximated by the learned exemplar views through a limited number of series of so-called radial basis functions (such as Gaussian functions) each of which becomes activated within a limited region of the high-dimensional feature space. Object recognition then means to examine whether a new point corresponding to the actual stimulus can be approximated by the existing tuned set of basis functions. Thus,

recognition does not occur by transformation or reconstruction of an internal image, but rather by interpolation or approximation of exemplars in a high-dimensional representational space.

At the end of the 1990s – and as an extension to the interpolation models – recognition models based on *pooling and thresholding* were developed [63.29–32]. Recognition is explained on the basis of the behavior of cells that are selectively tuned to specific image features (fragments or whole shapes) in a view-dependent (and size-dependent) way. A hierarchical pooling of the outputs of view-specific cells provides generalization over viewing conditions [63.29]. A similar proposal was made by Riesenhuber and Poggio [63.30]. The threshold model [63.33] also accounts for the systematic relation between recognition latencies and the amount of rotation (and size scaling): the speed of object recognition depends on the rate of accumulation of activity from neurons selective for the object, evoked by a particular viewing circumstance. For a familiar object, more tuned cells will be activated in the views most frequently presented, so that a given level of evidence (threshold) can be achieved fast. When the object is seen in an unusual view, fewer cells will respond, and activity among the population of cells selective for the object's appearance will accumulate more slowly. Consequently, these threshold models explain orientation dependency without the need to postulate transformation or interpolation processes.

Several computational models have been proposed that aim to model and explain the dependence of human recognition performance on spatial transformations. All of these models rely on storing exemplars – in the simplest form just 2-D views of objects – and matching the retinal image to these stored examples by different computational methods. The later models of recognition take their inspiration from recent findings from physiological studies concerning the functional building blocks of human vision in the brain. In the following, we will therefore briefly review the neural processing of visual information in the brain that underlies our ability to recognize objects.

63.2.2 Neural Representations in Object Recognition

Functionally, it has been shown that the flow of visual information in the brain can be divided into two major pathways: the *dorsal pathway* is believed to process motion and motor- or action-related visual information, whereas the *ventral pathway* is usually associated with

the task of object recognition. The structure of the ventral pathway is hierarchically organized and consists of a series of interconnected stages that start from the retina, passing through the lateral geniculate nucleus (LGN) to the primary visual cortex (V1) and extrastriate visual areas V2, V4, and IT. The *inferotemporal cortex* (IT) provides input to the *prefrontal cortex* (PFC), which is believed to play an important role in identification and categorization of visual stimuli. Recordings in the *parietal cortex* [63.34] suggest, in addition, that specifically for grasping and object manipulation dorsal regions might also be centrally involved in the recognition of manipulable objects and their affordances (see Chap. 62 for a more detailed discussion).

The seminal work of Hubel and Wiesel [63.35] in the cat (and later also in the macaque) visual cortex first established the idea of a hierarchical organization of visual processing. They found so-called simple cells in the early visual cortex (area V1) that responded best to bar-like stimuli at a particular orientation and position in the visual field. The response pattern of these cells could be modeled as a receptive field using Gabor-type functions. Later in the processing stream they found so-called complex cells which responded best to bar-like stimuli at a particular orientation nearly everywhere in the visual field: cells, which had become partially position invariant. This general idea of increasing invariance to stimulus properties with later stages of the processing stream has been verified in further physiological studies. In general, it has been found that the receptive field of the neurons increases and that the complexity of the stimulus it responds to increases. One of the key studies about the functional role of IT regions has investigated the responses of neurons to real-world objects in anesthetized monkeys ([63.36]; see also [63.37] for a review). Although some neurons that responded maximally to simple bar-like stimuli were found, the majority of neurons in the *posterior inferotemporal cortex* (PIT) preferred complex objects such as star shapes or circles with protruding elements. Interestingly, neurons were highly sensitive to minuscule changes to these objects such as the relative orientation or thickness of the elements. On the other hand, neurons were quite insensitive to stimulus variations such as size, contrast or retinal location. These findings were taken as evidence that one of the strategies for representing objects might be to use a number of moderately complex visual elements, whose pattern of coactivation encodes the visual appearance of the stimulus. In addition, Wang et al. [63.36] found neurons in the *anterior inferotemporal cortex* (AIT), that responded

maximally to images of whole objects such as faces or cars, indicating that already in the IT object-specific encodings might be present. Several other studies have also found neurons in this area which are tuned to faces, parts of faces, as well as body parts (see [63.38] for a review).

In another set of experiments, Logothetis et al. [63.39] found AIT neurons that showed a strong view-based behavior for the same stimuli that were used in the study of Bülthoff and Edelman [63.15], whereas they were invariant to the size and location of the stimulus. Their findings provide strong evidence that a neural implementation of view-based object encoding is possible and indeed seems to be used for recognition. In addition to view selectivity and size invariance, the investigated cells were also found to be maximally selective for the holistic stimulus rather than its constituent parts. This finding indicates that these cells might be encoding the pooled coactivation pattern of earlier PIT cells and thus form view-tuned units of recognition. It is important to stress in this context that an abstraction such as *grandmother* neurons, which specifically encode only one stimulus, does not seem plausible. Rather, the majority of neural responses in this and other experiments showed selectivity for a number of stimuli. A plausible explanation for this finding is that objects are encoded not by a single neuron but by a population code encompassing a number of neurons, which greatly increases the robustness of the representation (see also [63.32]).

The findings from this area of research can be summarized in a simple functional architecture: going from early stages to later stages of visual processing in a feedforward fashion, feature complexity increases from simple edge detectors towards view-tuned complex object cells and invariance to changes in the stimulus increases. This functional architecture is reflected not only in the object recognition framework discussed previously, but it also provides the motivation for computational vision systems that have been developed over the last few decades, which will be discussed in the following.

63.2.3 Object Recognition: Lessons from Computer Vision

Computer vision started out as a subfield of artificial intelligence in the 1960s. Early work on scene understanding by Roberts [63.40] showed how computers could *parse* worlds consisting of simple, geometric objects such as cubes, pyramids, etc. The main thrust of computer vision systems in the following decades

consisted of building algorithms for reconstructing a three-dimensional world from images – this development was further stimulated by Marr’s highly influential theory of vision as 3-D reconstruction [63.1]. This theory was built on extracting geometric primitives from images that could be mathematically described as generalized cylinders. Although the mathematical rigor of such approaches was very appealing, computational implementations turned out to have strong limitations. Extracting robust features is a necessary prerequisite for building a 3-D reconstruction of the image, and finding these features proved to be hard under real-world conditions due to the enormous amount of variation in the image caused by changes in lighting, depth rotations, noise, occlusion, etc.

Parallel to the paradigm shift in human psychophysics and physiology, exemplar-based computational systems began to emerge, which for the first time showed good recognition performance under a larger range of viewing conditions. These recognition systems were based on – sometimes surprisingly simple – histograms of pixel values [63.41], local feature detectors [63.42, 43] or on a straightforward pixel representation of images using principle components analysis [63.44]. All of these recognition systems relied on a database of labeled example images, an algorithm for extracting features from these images, and a suit-

able classification method for comparing sets of image features.

Returning to the discussion of modeling human vision, in the following we provide an exemplary review of three relatively recent recognition systems that are based on a functionally plausible, exemplar-based architecture: these are SpikeNET [63.45], LeNet [63.46], and a framework by *Serre et al.* [63.47]. The first system is motivated by the finding that humans are amazingly fast at categorizing images as containing an animal or a face [63.48]. Typical response times for this task are so small (on the order of 100 ms) that the visual signal only has time for one feedforward pass through the visual areas of the brain (Fig. 63.2) – any recurrent feedback processing would necessarily delay the decision and therefore result in longer response times. Based on this finding, a neural network architecture was designed [63.45] that exploits the timing of neuronal responses (spikes) to encode visual signals using a *who fires first* strategy. This is different from traditional neural networks in that the timing is used rather than the firing strength. An object in this system will therefore be represented by an ensemble of neurons that represents a pattern of spike responses from earlier low-level feature extraction neurons. In their implementation these low-level neurons consist of standard Gabor-type receptive fields that are similar to the receptive fields found in

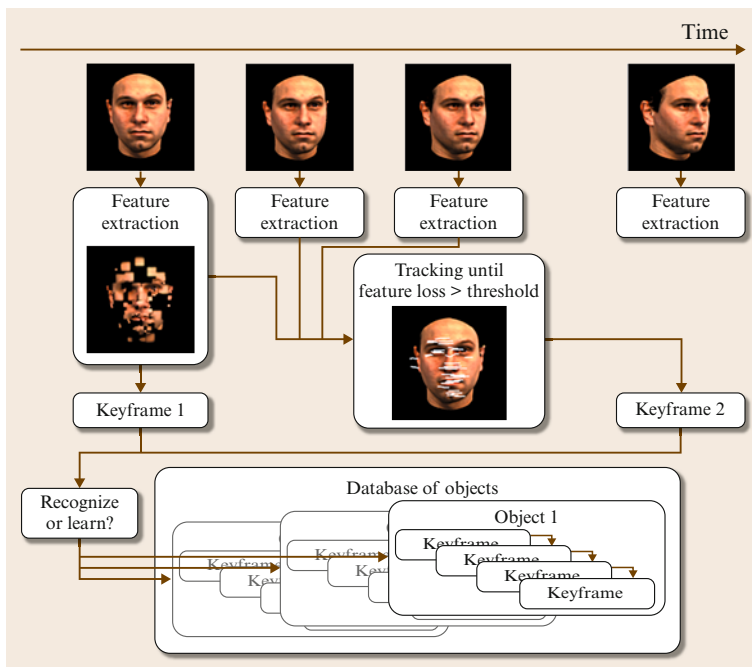


Fig. 63.2 The keyframe representation. This computational framework is a simple architecture that incorporates temporal input, incremental learning, structured object representations as well as efficient indexing. Note that this framework was also used for active exploration coupled with proprioceptive information (see text)

the cat's visual cortex [63.35]. This spike time encoding allows for very fast processing of visual stimuli and has been shown to provide robust recognition results. The network architecture LeNet [63.46] consists of a neural network that uses a hierarchy of layers of trainable convolutions and spatial subsampling, as well as non-linear filtering to extract features of increasingly large receptive fields, increasing complexity, and increasing robustness. Using extensive, supervised training of the full hierarchy, such a network provides a very efficient, sparse set of features for many visual recognition tasks. Finally, the network architecture by *Serre et al.* [63.47] uses a very similar hierarchical structure of layers in which feature complexity and invariance are successively increased by linear and nonlinear pooling; its lower-level feature detectors, however, are trained in an unsupervised fashion on a large database of natural images, yielding a large set of detectors that are optimally tuned to natural image statistics. Again, the performance of this model in recognition tasks has been shown to be very good; in addition, comparisons with physiological and psychophysical experiments have shown that this framework is also capable of modeling human results from these experiments. All of these systems have been demonstrated to work on real-world vision problems (such as recognition of objects in cluttered environments) and in the case of the system by *Serre et al.* [63.47] have been shown even to outperform other state-of-the-art computer vision algorithms. This would make them good candidates for building the visual learning and recognition system for a perceptual robot.

63.2.4 Object Learning and Recognition for Perceptual Robotics

In general, it can be said that the success of perceptually inspired recognition systems can be seen as a strong indicator for the feasibility of a data-driven exemplar-based approach to recognition. There are three issues, however, which so far have not been addressed in any of these vision systems and which will be important both for achieving human performance in generic recognition tasks in a perceptual robotics application – as well as for a full understanding of the processes in human object recognition.

First of all, all of the aforementioned systems are feedforward; virtually no feedback, recurrent processing is implemented in their architecture, which makes them in a sense very similar to the simpler frog- or bee-like neural systems discussed in Chap. 62. Although there

is evidence that humans solve some recognition tasks using very little feedback (see, for example, [63.48]), it is nevertheless a crucial component of visual processing driving, for example, attentional focus, context awareness, as well as memory and reasoning processes – basically everything that makes up visual *intelligence*. Some visual attention models that are relevant for robotics systems are reviewed in Chap. 62.

Secondly, a severe limitation of most of today's artificial recognition systems is that they solely focus on the static domain of object recognition. Visual input on the retina, however, consists of dynamic changes due to object- and self-motion, nonrigid deformations of objects, articulated object motion as well as scene changes such as variations in lighting, occluding and re- and disappearing objects, where at any given point in time several of these changes can be interacting. Several psychophysical experiments indeed suggest an important role for dynamic information, both in learning and recognition of objects [63.49–52]. These results ask for an extension of current object recognition frameworks with a temporal component in order to arrive at truly spatiotemporal object representations. Combining methods from computer vision, psychophysics, and machine learning, *Wallraven et al.* [63.53] developed a framework that fulfills this requirement and learns spatiotemporal exemplar-based object representations from image sequences (the so-called keyframe framework, Fig. 63.3). More specifically, spatiotemporal characteristics of the visual input are integrated into a connected view-graph representation based on tracked local features. In order to provide robust classification performance, machine learning techniques are used to design efficient methods for combining support vector classification schemes with these local feature representations [63.54]. In several studies it was shown that the framework achieved excellent recognition results on both highly controlled databases as well as on real-world data. The integration of spatiotemporal information provides characteristic information about dynamic visual input via the connection of views and the two-dimensional image motion of discriminative features. In addition to delivering good recognition performance, the framework was also able to model results from psychophysical experiments on face and object recognition.

A third issue that – in our view – will be essential for designing and implementing efficient perceptual robots consists of the multisensory nature of our perceptual system (see also the discussion of embodied robots in Chap. 56). As an example, there is a close coupling between the human visual and haptic system – touch can

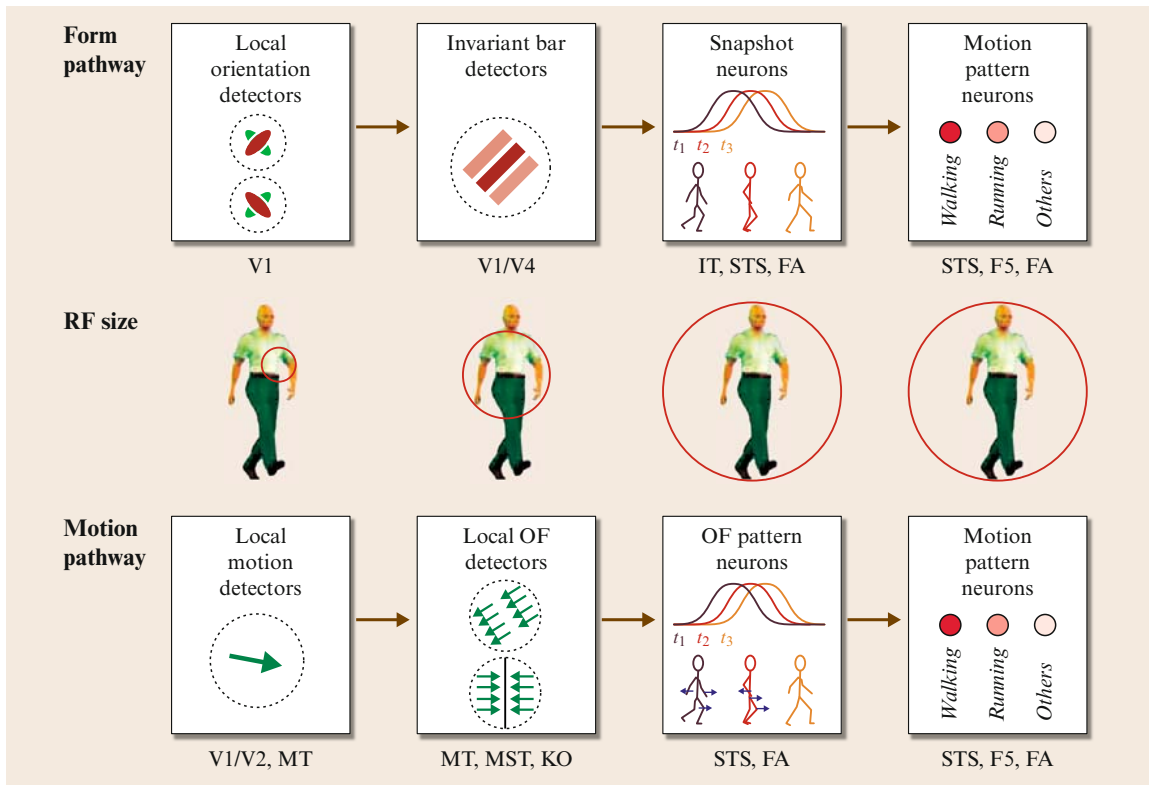


Fig. 63.3 Example-based neural model for the visual recognition of body movements that integrates the processing of form and motion features (after Giese and Poggio [63.56])

provide a wealth of complementary information about an object when it is manipulated, such as its texture, its shape, its position in space relative to our body, etc. In a series of psychophysical experiments (see [63.55]), participants had to learn views of four simple 3-D objects made of stacked toy bricks either in the haptic modality (when they were blindfolded) or in the visual modality (without being able to touch them). Subsequently they were tested both within the same modality as well as across modalities. Recognition results showed that cross-modal recognition is possible well above chance. Not surprisingly, recognition of rotated objects in the within-modality condition was severely affected by rotation in both modalities. This shows that not only is visual recognition highly view dependent but that haptic recognition performance is also directly affected by different viewing parameters. The results from this experiment thus support the view that haptic recognition is also mediated by exemplar-based processes.

Taken together with the keyframe framework outlined above, this cross-modal transfer might be an

important reason for the excellent visual performance of human object recognition – after all, it is known that infants learn extensively by grasping and touching objects, which could thus provide a *database* of object representations for visual recognition. Using this basic perceptual principle as a motivation, Wallraven and Bühlhoff [63.57] applied an extension of the keyframe framework in an online robotics scenario for efficient learning and recognition of multisensory object representations. More specifically, a framework was developed to integrate both *proprioceptive* information originating from haptic sensors in the robot's hands and *visual* information coming from the robot's cameras. For this, the robot would perform an exploratory movement with an object in its hand (such as turning it and looking at it from all angles) and from the resulting image sequence learn spatiotemporal view-based representations using the framework depicted in Fig. 63.2. Each view of this representation, however, is also linked to the current proprioceptive state (i. e., the joint angles of the hand at that point in time) and therefore provides an anchor into

a hand-centered three-dimensional space. In this way a representation is generated that *links perception and action*. The proprioceptive information can then be used

as an additional constraint for both learning of objects and recognition of objects and was shown to provide increased robustness compared to visual matching alone.

63.3 Example-Based Movement Representations

The recognition and modeling of complex movements and actions is fundamental to many applications in robotics, for example, interactive robots need to analyze users' movements in order to respond in a natural way to their behavior. This problem is specifically relevant for the realization of social and emotional behavior in robots (Chap. 58). Another important domain is the recognition and modeling of complex movements for humanoid robots (Chaps. 56 and 59) in order to learn new movements by imitation and to mimic human-like motor behavior.

63.3.1 Recognition of Complex Movements in Visual Cortex

The recognition of complex movements is also a fundamental problem for biological systems. While simple movement patterns are sufficient for eliciting stereotypical prey-catching behavior in simple vertebrates ([63.58]; see also Chap. 62), higher animals exploit more complex movement patterns, e.g., for the recognition of conspecifics or predators, or for communication by facial movements, gestures, or body expressions. Human perception of complex motion patterns is very efficient, even for extremely impoverished stimuli. This has been demonstrated in classical experiments by *Johansson* [63.59], who showed that complex dynamic actions can be recognized even from displays that consist of only a small number of dots moving like the joints of a human actor. Subsequent research has demonstrated that humans can extract highly specific information from such *point-light displays*, e.g., the gender or the identity of people. To our knowledge, no technical system for motion recognition has been proposed so far that accomplishes a comparable level of robustness. While much more research in neuroscience has been dedicated to object recognition (Sect. 63.2.2), some studies have tried to uncover neural [63.60–62] and computational principles [63.56, 63, 64] of visual movement recognition. Some of these principles seem interesting for the construction of robotics systems.

Neurophysiological and brain-imaging studies indicate that the recognition of facial and body movements

involves the ventral and the dorsal visual pathways. This implies that form and optic flow information are likely integrated during the processing of action stimuli in visual cortex. The ventral pathway, which is specifically responsible for the processing of form information, has already been discussed in Sect. 63.2.2. Like the ventral stream, the dorsal pathway is also hierarchically structured, and the size of the receptive fields of the neurons increases along the hierarchy. Some cortical areas that are part of the dorsal pathway are listed in Fig. 63.3. The *medial temporal area* (MT) contains neurons that are selective for simple local motion and coherent motion. On higher levels of the dorsal stream, e.g., in the *superior temporal sulcus* (STS), neurons that are selective for hand and body movements and for facial expressions have been found in monkeys and humans [63.62]. It seems likely that areas selective for human body shapes, e.g., the *extrastriate body part area* (EBA), contribute to the recognition of actions, even though some of them are not specifically selective for motion [63.65]. Finally, neurons that are active during action recognition have also been found in the *mirror neuron system*, which includes regions in the premotor and parietal cortex [63.66]. Parietal areas might also be specifically relevant for the recognition of action-related objects and their relationship to moving effectors [63.34]. The mirror neuron system has been very influential in recent research on the construction of biologically inspired robots (Chaps. 60 and 62).

A first biological principle of movement recognition that is supported by experimental data, and which seems useful for technical systems, is action recognition by detection of specific relevant motion and form features. Such detection does not necessarily require a reconstruction of facial or body shape, nor an exact simulation of the dynamics of the underlying movements. Instead it can be accomplished by much simpler feature detectors. Like object recognition, the recognition of complex motion patterns is strongly orientation and view dependent. This property has also been observed at the level of individual neurons in the STS, and for the activation of biological motion-selective areas in the human cortex [63.62]. View and orientation

dependence seem to be compatible with an encoding of visually perceived movements in terms of potentially learned example views or *keyframes* [63.56, 60]. While there might be some innate preferences for specific features, psychophysical and functional magnetic resonance imaging (fMRI) experiments suggest an important role of learning in visual movement recognition [63.67, 68]. For example, subjects can learn easily to recognize individually-specific body and facial movements [63.69, 70]. Also the fact that learning-based theoretical models, exploiting similar principles as neural object recognition models, can account for a variety of the experimental data [63.56, 71] supports a central role of learning.

As an example, Fig. 63.3 illustrates a hierarchical model for the recognition of complex movements that has been described in [63.56]. Its neural architecture consists of two hierarchical streams modeling the ventral and the dorsal visual pathways, which contain detectors for action-specific motion and form features. The form pathway of this model is similar to the object recognition models described in Sect. 63.2.2. The motion pathway of the model contains detectors for action-specific optic flow features with different complexity. Like for the described object recognition models, position and scale invariance is accomplished by appropriate nonlinear pooling of the responses of detectors with different spatial and scale selectivity along the hierarchy. In addition, the model contains recurrent neural circuits that make the responses of the recognition neurons selective for temporal order. In this way the model responds only to actions that are executed with the correct temporal order, and also with approximately the correct speed. The underlying network dynamics can be interpreted as a neural implementation of a Markov model, where the present recognized pattern predicts possible future patterns. A strong activity in the network emerges only when the stimulus sequence matches these predictions. A similar neural implementation exploiting the SPIKENET architecture by [63.45] (Sect. 63.2.3) has recently been applied in the context of a technical system for tracking of walkers in video sequences [63.72]. Hierarchical neural architectures inspired by the visual cortex have also been used in the context of mirror neuron robots [63.73].

A second principle of movement recognition that is important for the recognition of *imitable* actions, i. e., actions that can be executed by the observer, is an internal simulation of the observed motor behavior. While the exact nature of the underlying representations has not yet been clarified, a large body of experimental evidence suggests that observation of imitable movements acti-

vates the *mirror neuron system* in premotor and parietal cortex [63.66]. Computational models for action recognition by internal simulation have been proposed, e.g., exploiting feedforward controllers [63.74]. Theoretical models for the mirror neuron system with relevance for robotics are discussed in more detail in Chaps. 59 and 62.

63.3.2 Example-Based Motion Recognition in Robots and Computer Vision

Many systems for motion recognition and tracking in computer vision demonstrate the computational feasibility and applicability of the first principle: recognition by extraction of action-specific form and motion features without explicit modeling of the geometric shape of bodies or faces [63.75, 76]. In fact, most existing recent systems for face and body tracking are based on this approach. As opposed to the discussed neural models, many of these technical systems exploit nonbiological feature detectors and mechanisms for accomplishing temporal order selectivity and propagation of information over time (hidden Markov models (HMMs), and Kalman or particle filters [63.77–79]). Even though most of these solutions are not based on neural mechanisms, they are highly relevant in robotics for gesture recognition or the tracking of user behavior.

The second principle of action recognition by internal simulation, or by online fitting of geometrical or dynamical models, matches the traditional approach in computational vision, which had already been proposed by Marr [63.80]. A variety of implementations for model-based tracking and recognition of body movements [63.81] and facial movements [63.76] have been proposed in computer vision. Typically, such methods combine 2- or 3-D models, and sometimes even dynamic muscle models with predictive models such as Kalman or particle filters (for reviews see [63.77–79, 82]).

While in the biological system the exact nature of the representation of actions in the mirror neuron system is still quite unclear, it seems likely that the brain combines recognition of specific motion and form features and some form of internal simulation of motor programs for the recognition of imitable movements. Likewise, recent approaches in computer vision, computer graphics, and robotics have tried to combine feature-based recognition (e.g., of body silhouettes) and model-based tracking for the reconstruction of 2- or 3-D shape models from video streams [63.83, 84]. In fact, the combination of view-based recognition and model-based tracking improves the accuracy and robustness of body tracking [63.85].

63.4 Example-Based Synthesis Models: From Faces to Movements

In Sects. 63.2 and 63.3 biologically inspired approaches to the example-based recognition of shapes and movement patterns were discussed. For biological systems it seems likely that the visual recognition of complex patterns and the synthesis of complex patterns in motor control share fundamental principles, like example-based representations that are capable of efficient generalization [63.86]. Referring back to the most abstract level of analogies between biological and technical systems, example-based models have also been applied for the synthesis of complex patterns in technical applications, such as highly realistic face pictures and movements. These applications are highly relevant for robotics since they form the basis of the simulation of human-like behavior with high believability. The increasing importance of humanoid robots (Chaps. 56, 58, and 59) and of robotics systems that can interface with normal human social behavior (Chap. 58) makes believable simulation of human behavior a central topic in modern robotics. In several cases, example-based learning techniques have been superior to classical methods based on geometrical or physical models for such applications. A further application domain for example-based synthesis is the automatic generation of virtual training examples for learning-based robotics (Chap. 59) or computer vision systems [63.87].

63.4.1 Face Synthesis: Realistic Faces for Avatars and Robots

One of the crucial ingredients of careful studies of face processing – both for facial expressions and identity recognition – is a highly controllable, yet at the same time realistic stimulus set. While it is certainly possible to do research with impoverished or highly abstracted stimulus sets (such as line drawings) or only in the static domain (the majority of research has been done using static photographs), the results are certainly limited in terms of their generalizability to the real world. One of the recent thrusts in cognitive research has therefore been to use highly realistic face renderings – both static and dynamic – using advanced computer graphics techniques. As this technology also offers novel insights in the context of robotics, we will briefly describe a high-end facial animation system that was used in several studies. One important application domain of synthetic faces is for communication and interaction in any scenario in which communication between a robot and a human is required. Being able to synthe-

size realistic faces allows the robot to communicate its internal state and intentions in a manner that is very natural and easy to understand for humans. In addition, this research can be used not only for synthesis purposes but also for analysis purposes, that is, to design vision systems for robots that understand and react to the complexity of human faces appropriately. Again, we envision such a recognition and analysis system to be an essential component of any robotic application that relies on interactions with humans, especially in the field of anthropomorphic robotics.

The idea of a face space – in which faces are represented in a high-dimensional vector space – was pioneered by Valentine in 1991 and has been influential especially in perceptual research on face processing [63.88]. A number of techniques in computer vision and computer graphics have implemented the concept of a face space, first by interpolation or morphing between example pictures [63.89, 90]. As a further step towards the simulation of highly realistic pictures of faces, Blanz and Vetter [63.91] created a three-dimensional face space from 3-D laser scans of faces. The critical step for defining a vector space of faces was to bring the example patterns into correspondence. For 3-D laser scans this means that, for example, the tip of the nose can be uniquely identified in all scans. By using a modified optic flow algorithm, which establishes a dense correspondence between two 3-D scans, Blanz and Vetter were able to create such a face space. Such vector space models generated by interpolation between example pictures (or scans) that are in spatial correspondence are also called *morphable models* (Fig. 63.4). Based on such models the properties of face images, e.g., their male- or femaleness, can be continuously varied. This allows the synthesis of artificial new facial expressions, or of face pictures with novel identities. Also existing pictures of faces can be modified, resulting in novel pictures with photorealistic quality [63.91].

By analysis of the statistical properties of face spaces, perception researchers have been able to conduct highly controlled experiments, e.g., addressing gender perception [63.92] or identity coding [63.93]. In addition, an encoding of face stimuli in terms of a vector space seems to be compatible with recent psychophysical results in humans [63.94], and electrophysiological results in monkeys [63.95]. This makes this concept interesting as a basis for the design of perceptual robots that have to synthesize and analyze faces (Chap. 58).

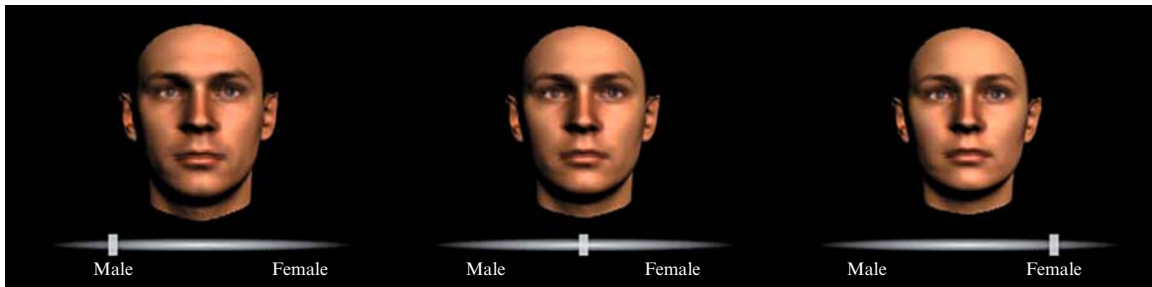


Fig. 63.4 Morphable model: the three images show a morph from an average male face to an average androgynous face to an average female face

Recently, static morphable models were extended as the basis of a dynamic animation system [63.96]. The basic idea is to combine the high spatial resolution of 3-D laser scans with the high temporal resolution of motion capture. Facial geometry is first captured using a structured light scanner, which produces a detailed 3-D scan of each peak expression. The individual scans are put into correspondence with each other based on a control mesh whose 3-D fiducial points are aligned to each scan, which results in a morphable model in much the same way as the one described earlier. The second step in creating the animation system is to capture motion data with an optical motion capture system. On this motion capture data, simple linear detectors for facial action elements are specified by using distances between markers to drive the expression activation. This yields a realistic animation based on the amplitude and timing of marker motion in the motion capture data. The morph animations of the final animation system thus allow full control over important information channels used in facial expressions (such as internal motion of the face, rigid head motion, or fidelity of shape and texture), making it ideally suited for experiments on perception of facial expressions [63.97]. The results from such experiments could, for example, be used in perceptual robotics to design robots that are able to understand (being able to analyze what is said and meant by a human) and react (being able to synthesize a realistic facial expression) to the variety of human facial expressions in a realistic fashion thus enabling lifelike interaction with users.

63.4.2 Example-Based Trajectory Synthesis

Example-based techniques have been successful for the synthesis of photorealistic pictures (Sect. 63.4.1) and the recognition of complex movements (Sect. 63.3.1). It turns out that related methods are also suitable for the

modeling of complex movement trajectories by interpolation between learned examples. This approach has been very successful in computer graphics, specifically for the modeling of highly realistic human movements. The simulation of realistic human movements for platforms with many degrees of freedom, like avatars and humanoid robots, based on physical or dynamical models is extremely difficult. Learning-based techniques exploiting motion capture data have thus become the most successful approach for the high-quality synthesis of complex human movements. A variety of methods for the synthesis of movement trajectories by interpolation and editing of motion-captured trajectory segments have been developed in computer graphics. Many of these approaches are also interesting for movement synthesis in robotics.

Simple approaches for example-based trajectory modeling by interpolation define novel movement styles by *motion morphing* between presegmented trajectories [63.98,99]. An example is the interpolation between different gait styles. A variety of methods have been proposed for motion morphing (for a review see [63.100]), i. e., the blending between different movements styles. Some of these methods define linear Euclidean-style spaces by linear combination of example trajectories [63.83, 101, 102]. Such style spaces can be seen in analogy to face spaces. Linear combination methods for trajectories thus represent a spatiotemporal equivalent of morphable models (see Sect. 63.4.1). Another class of methods interpolate between example trajectories by learning of generative statistical models for the space-time texture of the training patterns. Such methods often yield real-time capable stochastic models for the generation of novel trajectories, e.g., exploiting hidden Markov models [63.103]. Real-time capability is essential for robotics applications, since usually it is not possible to presynthesize complete movements offline. Instead, robotic systems need to be reactive and

capable of online correction of synthesized movements depending on changes in the environment.

While simple motion interpolation techniques require presegmented and aligned example trajectories, more-advanced techniques have been proposed that accomplish an automatic segmentation of complex action streams into simpler components, or *movement primitives*. The concept of movement primitives has been quite influential in the theory of motor control of biological systems. A variety of different definitions of primitives has been given in the literature of motor control, ranging from coordinated activation patterns (synergies) at the muscle level to more abstract cognitive units of movement planning (for a review see [63.104]). In the context of robotics different approaches have been taken to identify and implement movement primitives. Sometimes, primitives have been predefined heuristically, e.g., as single goal-directed or periodic movements. Recent studies have tried to learn primitives directly from trajectory data sets applying supervised or unsupervised learning techniques, for example principle components analysis (PCA) or ISOMAP [63.105, 106]. Temporal correspondence between learned primitives and movements novel sequences can be established by appropriate dynamic alignment algorithms [63.107]. Optimized transitions between prestored or synthesized trajectory segments can be generated automatically, e.g., exploiting graph optimization algorithms [63.108, 109]. In this way, trajectories for complex action sequences can be synthesized automatically from learned example trajectories, ensuring that they fulfill specific style properties and are consistent with additional kinematic or dynamic constraints.

Some studies have embedded such learned trajectory models into real-time capable systems, e.g., for online tracking and synthesis of movement trajectories inter-

active applications like computer games [63.101, 110]. Such methods seem highly suitable for online imitation of movements in humanoid robots. Finally, movement primitives can also be modeled as dynamical control policies. It has been shown that such policies can be encoded by nonlinear dynamical systems whose stable solutions reproduce complex training movements [63.111] (see Chap. 59 for details).

Movements generated by interpolation between example trajectories can violate constraints of robot systems, e.g., the kinematic or dynamic limits of the platform and the environment. One possibility is to use the synthesized movement as a *virtual trajectory* that specifies a temporal sequence of goal states for the robot controller [63.107]. A variety of *filtering techniques* have been proposed that deform the synthesized trajectories in a way that makes them compatible with additional spatial or temporal constraints [63.112]. By exploiting physical models compatibility with dynamic constraints, like the minimization of torques, can also be accomplished [63.106]. For walking avatars and robots additional dynamic stability constraints have to be taken into account, e.g., concerning the dynamic stability of walking. Methods for deforming synthesized trajectories to make them compatible, for example with the zero moment point (ZMP) criterion (Chap. 16), have been proposed [63.113]. Such approaches try to combine the advantages of example-based trajectory generation (high degrees of accuracy and realism) with the ones of classical physics-based trajectory planning (real-time capability, simplicity of expressing physical and task constraints). However, the optimal integration of efficient learning models of complex movements and classical concepts in robot control is an active field of research in learning-based and humanoid robotics (Chap. 59).

63.5 Conclusions and Further Reading

In this chapter we have presented several principles derived from high-level cognitive processing in vision in the human brain that have been fruitful for the development of systems in robotics and computer vision. The recognition of shapes and complex movements and actions is an important problem for many applications in robotics. We have discussed a variety of results from neuroscience that indicate that in the brain these functions are likely realized by example-based representations. We have discussed the neural implementations

of such representations, which have been tested in the context of technical applications and which are strongly inspired by the real cortical neural architecture. In addition, we have discussed a class of methods for the highly accurate modeling of complex shapes and movement trajectories that are also based on learned examples, but which exploit more efficient technical algorithms for interpolation between the example patterns. We finally discussed approaches that try to link such example-based representations to more traditional model-based

recognition and tracking approaches. In addition, we have mentioned some methods for adapting patterns that are synthesized by such example-based techniques to boundary conditions and constraints of real robot platforms, and to accomplish real-time capability. We think that the discussed spectrum of results and applications shows that a transfer between biological and robotics systems on multiple levels can help to develop more efficient technical systems.

Example-based mechanisms for object and motion recognition account for the invariant recognition of complex patterns. However, they do not automatically extract the metric information about the object geometry and position and the spatial parameters of complex trajectories in world coordinates. For some tasks in robotics, such as grasping, manipulation or obstacle avoidance, such information is required (see, e.g., Chaps. 28 and 52). For such tasks, example-based recognition must be fused with methods for the extraction of the relevant metric information. In robotics such information can be extracted by stereo vision or using special sensors such as laser range finders. In the brain the fusion between such spatial information and information about objects likely occurs in the parietal areas, such as the *anterior interparietal area* (AIP) [63.114]. However, it is unclear whether the information about objects is only represented in terms of 2-D example views. Instead, it seems likely that some form of 3-D information is also encoded, potentially in an example-based manner. Also haptic and visual information about object shape might be merged in higher brain areas, e.g., in the parietal

and fusiform areas [63.115]. A more detailed discussion of biologically inspired models for the extraction of action-relevant geometrical information of objects and its combination with spatial information for grasping is presented in the context of the mirror neuron system.

In Sect. 63.2.1 we contrasted example- and model-based theories of the representation of objects in the brain. In fact, in a recent paper an attempt was made to reconcile these two approaches to object processing [63.116]: a careful study of the view dependency of novel objects was designed by combining structural properties (number of parts) with metric properties (thickness, size of parts) and found that both view-dependent and view-independent processing seem to be combined in object recognition. Thus, instead of taking the extreme standpoints of view-based versus view-invariant processing, one might envisage a visual processing framework in which features are selected according to the current task, where the optimality, efficiency, and thus the dependency on viewing parameters of the features depend on the amount of visual experience with this particular task. This suggests that a similar flexible approach might be suitable for the solution of problems in robotics. The combination of metric model-based representations for the planning of goal-directed movements and of example-based representations for the recognition of objects, people, and actions might result in powerful robotics systems with performance levels more similar to biological systems than the presently available solutions in robotics.

References

- 63.1 D. Marr: *Vision* (Freeman, San Francisco 1982)
- 63.2 R.A. Brooks: A robust layered control system for a mobile robot, *IEEE J. Robot. Automat.* **2**(1), 14–23 (1986)
- 63.3 J.A.S. Kelso: *Dynamic Patterns: The Self-Organization of Brain and Behaviour* (MIT Press, Cambridge 1995)
- 63.4 G. Schöner, M. Dose, C. Engels: Dynamics of behavior: theory and applications for autonomous robot architectures, *Robot. Auton. Syst.* **16**, 213–245 (1997)
- 63.5 J. Tani, M. Ito: Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment, *IEEE Trans. Syst. Man Cybernet. Part A: Syst Humans* **33**(4), 481–488 (2003)
- 63.6 W.H. Warren: The dynamics of perception and action, *Psychol. Rev.* **113**, 358–389 (2006)
- 63.7 D. Marr, H. Nishihara: Representation and recognition of the spatial organization of three-dimensional shapes, *Proc. R. Soc. London B* **200**, 269–294 (1978)
- 63.8 I. Biederman: Recognition-by-components: A theory of human image understanding, *Psycholog. Rev.* **94**, 115–147 (1987)
- 63.9 D. Lowe: *Perceptual Organization and Visual Recognition* (Kluwer, Boston 1985)
- 63.10 S. Ullman: *High-level vision. Object Recognition and Visual Cognition* (MIT Press, Cambridge 1996)
- 63.11 M.A. Kurbat: Structural description theories: is RBC/IIM a general-purpose theory of human entry-level object recognition?, *Perception* **23**, 1339–1368 (1994)
- 63.12 S. Edelman: *Representation and Recognition in Vision* (MIT Press, Cambridge 1999)
- 63.13 M.J. Tarr, H.H. Bülthoff: Image-based object recognition in man, monkey and machine, *Cognition* **67**(1–2), 1–20 (1998)

- 63.14 M. Graf, W. Schneider: Structural descriptions in HIT – a problematic commitment, *Behav. Brain Sci.* **24**, 483–484 (2001)
- 63.15 H.H. Bülthoff, S. Edelman: Psychophysical support for a two-dimensional view interpolation theory of object recognition, *Proc. Nat. Acad. Sci. USA* **89**, 60–64 (1992)
- 63.16 M.J. Tarr, S. Pinker: Mental orientation and orientation-dependence in shape recognition, *Cognit. Psychol.* **21**, 233–282 (1989)
- 63.17 W.G. Hayward, M.J. Tarr: Testing conditions for viewpoint invariance in object recognition, *J. Exp. Psychol. Human Percept. Perform.* **23**, 1511–1521 (1997)
- 63.18 S.E. Palmer, E. Rosch, P. Chase: Canonical perspective and the perception of objects. In: *Attention and Performance IX*, ed. by J. Long, A. Baddeley (Erlbaum, Hillsdale 1981) pp.135–151
- 63.19 H. Hill, P.G. Schyns, S. Akamatsu: Information and viewpoint dependence in face recognition, *Cognition* **62**, 201–222 (1997)
- 63.20 C. Wallraven, A. Schwaninger, S. Schuhmacher, H.H. Bülthoff: View-Based Recognition of Faces in Man and Machine: Re-visiting Inter-Extra-Ortho, *Lect. Notes Comput. Sci.* **2525**, 651–660 (2002)
- 63.21 M.J. Tarr: Rotating objects to recognize them: a case study on the role of viewpoint dependency in the recognition of three-dimensional objects, *Psychonom. Bull. Rev.* **2**, 55–82 (1995)
- 63.22 R. Lawson, G.W. Humphreys: View-specific effects of depth rotation and foreshortening on the initial recognition and priming of familiar objects, *Percept. Psychophys.* **60**, 1052–1066 (1998)
- 63.23 E. Ashbridge, D.I. Perrett: Generalizing across object orientation and size. In: *Perceptual Constancy. Why Things look as they do*, ed. by V. Walsh, J. Kulikowski (Cambridge University Press, Cambridge 1998) pp.192–209
- 63.24 M. Dill, S. Edelman: Imperfect invariance to object translation in the discrimination of complex shapes, *Perception* **30**, 707–724 (2001)
- 63.25 K.R. Cave, S. Pinker, L. Giorgi, C.E. Thomas, L.M. Heller, J.M. Wolfe, H. Lin: The representation of location in visual images, *Cognit. Psychol.* **26**, 1–32 (1994)
- 63.26 S. Ullman: Aligning pictorial descriptions: an approach to object recognition, *Cognition* **32**, 193–254 (1989)
- 63.27 S. Ullman, R. Basri: Recognition by linear combinations of models, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**, 992–1006 (1991)
- 63.28 T. Poggio, S. Edelman: A network that learns to recognize three-dimensional objects, *Nature* **343**, 263–266 (1990)
- 63.29 D. Perrett, W.M. Oram: Visual recognition based on temporal cortex cells: viewer-centred processing of pattern configurations, *Zeitschrift Naturforschung C* **53**, 518–541 (1998)
- 63.30 M. Riesenhuber, T. Poggio: Hierarchical models of object recognition in cortex, *Nature Neurosci.* **2**, 1019–1025 (1999)
- 63.31 E.T. Rolls, T. Milward: A model of invariant object recognition in the visual system: learning rules, activation functions, lateral inhibition, and information-based performance measures, *Neural Comput.* **2**(11), 2547–2572 (2000)
- 63.32 G. Wallis, H.H. Bülthoff: Learning to recognize objects, *Trends Cognit. Sci.* **3**, 22–31 (1999)
- 63.33 D. Perrett, W.M. Oram, E. Ashbridge: Evidence accumulation in cell populations responsive to faces: an account of generalization of recognition without mental transformations, *Cognition* **67**, 111–145 (1998)
- 63.34 H. Sakata: The role of the parietal cortex in grasping, *Adv. Neurol.* **93**, 121–139 (2003)
- 63.35 D.H. Hubel, T.N. Wiesel: Receptive fields, binocular interaction and functional architecture in the cat's visual cortex, *J. Physiol. (London)* **160**, 106–154 (1962)
- 63.36 G. Wang, M. Tanifuji, K. Tanaka: Functional architecture in monkey inferotemporal cortex revealed by in vivo optical imaging, *Neurosci. Res.* **32**, 33–46 (1998)
- 63.37 K. Tanaka: Representation of visual feature objects in the inferotemporal cortex, *Neural Netw.* **9**(8), 1459–1475 (1996)
- 63.38 K. Grill-Spector, R. Malach: The human visual cortex, *Annu. Rev. Neurosci.* **27**, 649–677 (2004)
- 63.39 N.K. Logothetis, J. Pauls, H.H. Bülthoff, T. Poggio: View-dependent object recognition by monkeys, *Curr. Biol.* **4**, 401–414 (1994)
- 63.40 L. Roberts: Machine perception of three-dimensional solids. In: *Optical and Electro-optical Information Processing*, ed. by J.T. Tippet (MIT Press, Cambridge 1965) pp.159–197
- 63.41 M. Swain, D. Ballard: Color indexing, *Int. J. Comput. Vis.* **7**, 11–32 (1991)
- 63.42 C. Schmid, R. Mohr: Local greyvalue invariants for image retrieval, *IEEE Trans. Pattern Mach. Intell.* **19**, 530–535 (1997)
- 63.43 D. Lowe: Distinctive image features from scale invariant keypoints, *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
- 63.44 M. Kirby, L. Sirovich: Applications of the Karhunen-Loeve procedure for the characterization of human faces, *IEEE Trans. Pattern Mach. Intell.* **12**, 103–108 (1990)
- 63.45 A. Delorme, S. Thorpe: SpikeNET: An event-driven simulation package for modeling large networks of spiking neurons, *Netw. Comput. Neural Syst.* **14**, 613–627 (2003)
- 63.46 Y. LeCun, F. Huang, L. Bottou: Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting in *Proceedings of 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2004)

- 63.47 T. Serre, L. Wolf, T. Poggio: Object recognition with features inspired by visual cortex. In: *Proceedings of 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2005)
- 63.48 S. Thorpe, D. Fize, C. Marlot: Speed of processing in the human visual system, *Nature* **381**(6582), 520–522 (1996)
- 63.49 G. Wallis, H.H. Bülthoff: Effects of temporal association on recognition memory, *Proc. Nat. Acad. Sci. USA* **98**, 4800–4804 (2001)
- 63.50 J.V. Stone: Object recognition using spatio-temporal signatures, *Vis. Res.* **38**, 947–951 (1998)
- 63.51 J.V. Stone: Object recognition: View-specificity and motion-specificity, *Vis. Res.* **39**, 4032–4044 (1999)
- 63.52 Q.C. Voung, M.J. Tarr: Rotation direction affects object recognition, *Vis. Res.* **44**(14), 1717–1730 (2004)
- 63.53 C. Wallraven, H.H. Bülthoff: Automatic acquisition of exemplar-based representations for recognition from image sequences. *CVPR 2001 – Workshop on Models versus Exemplars* (2001)
- 63.54 C. Wallraven, B. Caputo, A.B.A. Graf: Recognition with local features: The kernel recipe, *Proc. Int. Conf. Comput. Vis.* **2**, 257–264 (2003), IEEE Press
- 63.55 F.N. Newell, M.O. Ernst, B.S. Tjan, H.H. Bülthoff: Viewpoint dependence in visual and haptic object recognition, *Psychol. Sci.* **12**, 37–42 (2001)
- 63.56 M.A. Giese, T. Poggio: Neural mechanisms for the recognition of biological movements, *Nat. Rev. Neurosci.* **4**, 179–192 (2003)
- 63.57 C. Wallraven, H.H. Bülthoff: *Object Recognition in Man and Machine. Object Recognition, Attention and Action* (Springer, Tokyo) (in press)
- 63.58 J.-P. Ewert: Neural mechanisms of prey-catching and avoidance behavior in the toad (*bufo bufo*), *Brain Behav. Evol.* **3**, 36–56 (1970)
- 63.59 G. Johansson: Visual perception of biological motion and a model for its analysis, *Percept. Psychophys.* **14**, 201–211 (1973)
- 63.60 K. Verfaillie: Perceiving human locomotion: priming effects in direction discrimination, *Brain Cogn.* **44**, 192–213 (2000)
- 63.61 A.J. O'Toole, D.A. Roark, H. Abdi: Recognizing moving faces: A psychological and neural synthesis, *Trends Cognit. Sci.* **6**, 261–266 (2002)
- 63.62 D. Perrett, A. Puce: Electrophysiology and brain imaging of biological motion, *Phil. Trans. R. Soc. Lond. B* **358**, 435–445 (2003)
- 63.63 D.D. Hoffman, B.E. Flinchbaugh: The interpretation of biological motion, *Biol. Cybernet.* **42**, 195–204 (1982)
- 63.64 J.A. Webb, J.K. Aggarwal: Structure from motion of rigid and jointed objects, *Artif. Intell.* **19**, 107–130 (1982)
- 63.65 M. Peelen, A. Wiggett, P. Downing: Patterns of fMRI activity dissociate overlapping functional brain areas that respond to biological motion, *Neuron* **49**, 815–822 (2006)
- 63.66 G. Rizzolatti, L. Craighero: The mirror-neuron system, *Ann. Rev. Neurosci.* **27**, 169–192 (2004)
- 63.67 E.D. Grossman, R. Blake, C.Y. Kim: Learning to see biological motion: brain activity parallels behavior, *J. Cogn. Neurosci.* **16**, 1669–1679 (2004)
- 63.68 J. Jastorff, Z. Kourtzi, M.A. Giese: Learning to discriminate complex movements: Biological versus artificial trajectories, *J. Vis.* **6**, 791–804 (2006)
- 63.69 H. Hill, F.E. Pollick: Exaggerating temporal differences enhances recognition of individuals from point light displays, *Psychol. Sci.* **11**, 223–228 (2000)
- 63.70 B. Knappmeyer, I.M. Thornton, H.H. Bülthoff: The use of facial motion and facial form during the processing of identity, *Vision Res.* **43**, 1921–1936 (2003)
- 63.71 J. Lee, W. Wong: A stochastic model of coherent motion detection, *Biol. Cybernet.* **91**, 306–314 (2004)
- 63.72 P. Kornprobst, T. Vieille, I.K. Dima: Could early visual processes be sufficient to label motions? *Proceedings of the IEEE International Joint Conference of Neural Networks (IJNN'05)* (2005) pp. 1687–1692
- 63.73 G. Metta, G. Sandini, L. Natale, L. Craighero, L. Fadiga: Understanding mirror neurons: a bio-robotic approach, *Interaction Studies* **7**, in press
- 63.74 D.M. Wolpert, K. Doya, M. Kawato: A unifying computational framework for motor control and social interaction, *Philos. Trans. R. Soc. London B Biol. Sci.* **358**, 593–602 (2003)
- 63.75 A. Bobick: Movement, activity, and action: The role of knowledge in the perception of motion, *Philos. Trans. R. Soc. London B* **352**, 1257–1265 (1997)
- 63.76 I. Essa, A. Pentland: Coding, analysis, interpretation, and recognition of facial expressions, *IEEE Trans. Pattern Anal. Mach. Intell.* **19**, 757–763 (1997)
- 63.77 J.K. Aggarwal, Q. Cai: Human motion analysis: A review, *Comput. Vision Image Understand.* **73**, 428–440 (1999)
- 63.78 D.M. Gavrila: The visual analysis of human movement: A survey, *Comp. Vis. Image Underst.* **73**, 82–98 (1999)
- 63.79 T.B. Moeslund, G. Granum: A survey of computer vision-based human motion capture, *Comp. Vis. Image Underst.* **81**, 231–268 (2001)
- 63.80 D. Marr, L.M. Vaina: Representation and recognition of the movements of shapes, *Proc. R. Soc. London B* **214**, 501–524 (1982)
- 63.81 S. Wachter, H.-H. Nagel: Tracking of persons in monocular image sequences, *Comput. Vis. Image Underst.* **74**, 174–192 (1999)
- 63.82 J.J. Wang, S. Singh: Video analysis of human dynamics – a survey, *Real-Time Imaging* **9**, 321–346 (2003)
- 63.83 R. Rosales, S. Sclaroff: Inferring body pose without tracking body parts, *Proc. IEEE Conf. Comput. Vis. Pattern Recognition (CVPR)* **II**, 721–727 (2000)

- 63.84 A. Agarwal, B. Triggs: Recovering 3D human pose from monocular images, *IEEE Trans. Pattern Anal. Machine Intell. (PAMI)* **28**, 44–58 (2006)
- 63.85 C. Curio, M.A. Giese: Combining View-Based and Model-Based Tracking of Articulated Human Movements. In: *IEEE Workshop on Applications of Computer Vision/IEEE Workshop on Motion and Video Computing (WACV/MOTION 2005)* (IEEE Computer Society, Los Alamitos 2005) pp. 261–268
- 63.86 T. Poggio, E. Bizzi: Generalization in vision and motor control, *Nature* **431**, 768–774 (2004)
- 63.87 P. Niyogi, F. Giorzi, T. Poggio: Incorporating prior information in machine learning by creating virtual examples, *Proc. IEEE* **86**, 2196–2209 (1998)
- 63.88 T. Valentine: A unified account of the effects of distinctiveness, inversion, and race in face recognition, *Quart. J. Exp. Psychol. A: Human Exp. Psychol.* **43**, 161–204 (1991)
- 63.89 T. Vetter, T. Poggio: Linear object classes and image synthesis from a single example image, *IEEE Trans. Pattern Anal. Mach. Intell.* **19**, 733–742 (1997)
- 63.90 A. Lanitis, C.J. Taylor, T.F. Cootes: Automatic Interpretation and Coding of Face Images Using Flexible Models, *IEEE Trans. Pattern Mach. Intell.* **19**, 743–756 (1997)
- 63.91 V. Blanz, T. Vetter: A morphable model for the synthesis of 3d faces. In: *Proc. ACM SIGGRAPH* (1999) pp. 187–194
- 63.92 I. Bülthoff, F.N. Newell: Categorical perception of sex occurs in familiar but not unfamiliar faces, *Vis. Cognit.* **11**, 823–855 (2004)
- 63.93 A.J. O’Toole, T. Vetter, V. Blanz: Three-dimensional shape and two-dimensional surface reflectance contributions to face recognition: An application of three-dimensional morphing, *Vis. Res.* **39**, 3145–3155 (1999)
- 63.94 D.A. Leopold, A.J. O’Toole, T. Vetter, V. Blanz: Prototype-referenced shape encoding revealed by high-level after effects, *Nat. Neurosci.* **4**, 89–94 (2001)
- 63.95 D.A. Leopold, I.V. Bondar, M.A. Giese: Norm-based face encoding by single neurons in the monkey inferotemporal cortex, *Nature* **442**, 572–575 (2006)
- 63.96 M. Breidt, C. Wallraven, D.W. Cunningham, H.H. Bülthoff: *Facial Animation Based on 3D Scans and Motion Capture. SIGGRAPH ’03 Sketches & Applications* (ACM, New York 2003)
- 63.97 C. Wallraven, M. Breidt, D.W. Cunningham, H.H. Bülthoff: Psychophysical evaluation of animated facial expressions, *Proc. 2nd Symp. Appl. Percept. Graphics Visual.*, 17–24 (2005)
- 63.98 M. Unuma, K. Anjyo, R. Takeuchi: Fourier principles for emotion-based human figure animation, *ACM Trans. Comput. Graphics* **29**, 91–99 (1995)
- 63.99 A. Bruderlin, L. Williams: Motion signal processing, *ACM Trans. Comput. Graphics* **29**, 97–104 (1995)
- 63.100 D.J. Wiley, J.K. Hahn: Interpolation synthesis of articulated figure motion, *IEEE Comp. Graphics Appl.* **17**, 39–45 (1997)
- 63.101 Y. Yacoob, M.J. Black: Parameterized modeling and recognition of activities, *Comput. Vis. Image Understand.* **73**, 232–247 (1999)
- 63.102 M.A. Giese, T. Poggio: Morphable models for the analysis and synthesis of complex motion patterns, *Int. J. Comput. Vis.* **38**, 59–73 (2000)
- 63.103 M. Brand, A. Hertzmann: *Style Machines. Proc. ACM SIGGRAPH 2000* (ACM, New York 2000) pp. 183–192
- 63.104 T. Flash, B. Hochner: Motor primitives in vertebrates and invertebrates, *Curr. Opin. Neurobiol.* **15**(6), 660–666 (2005)
- 63.105 O.C. Jenkins, M.J. Mataric: Performance-derived behavior vocabularies: Data-driven acquisition of skills from motion, *Int. J. Human Robot.* **1**, 237–288 (2004)
- 63.106 A. Safonova, J.K. Hodgins, N.S. Pollard: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces, *ACM Trans. Comput. Graphics* **23**, 514–521 (2004)
- 63.107 W. Ilg, G.H. Bakir, J. Mezger, M.A. Giese: On the representation, learning and transfer of spatio-temporal movement characteristics, *Int. J. Human Robot.* **1**, 613–636 (2004)
- 63.108 O. Arikan, D.A. Forsyth, J.F. O’Brien: Motion synthesis from annotations, *ACM Trans. Graphics* **22**, 402–408 (2003)
- 63.109 L. Kovar, M. Gleicher, F. Pighin: *Motion Graphs. Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques* (ACM, New York 2002) pp. 473–482
- 63.110 L. Ren, G. Shakhnarovich, J.K. Hodgins, H. Pfister, P. Viola: Learning silhouette features for control of human motion, *ACM Trans. Graphics* **24**, 1303–1331 (2005)
- 63.111 S. Schaal, A. Ijspeert, A. Billard: Computational approaches to motor learning by imitation, *Phil. Trans. R. Soc. London B* **358**, 537–547 (2003)
- 63.112 M. Gleicher, P. Litwinowicz: Constraint-based motion adaptation, *J. Vis. Comput. Anim.* **9**, 65–94 (1998)
- 63.113 S. Tak, H.S. Ko: A physically based motion retargeting filter, *ACM Trans. Graphics* **24**, 98–117 (2005)
- 63.114 C.L. Colby: Action-oriented spatial reference frames in cortex, *Neuron* **20**, 15–24 (1998)
- 63.115 A.R. Kilgour, R. Kitada, P. Servos, T.W. James, S.J. Lederman: Haptic face identification activates ventral occipital and temporal areas: an fMRI study, *Brain Cogn.* **59**, 246–257 (2005)
- 63.116 D.H. Foster, S.J. Gilson: Recognizing novel three-dimensional objects by summing signals from parts and views, *Proc. R. Soc. London B* **269**, 1939–1947 (2002)

64. Roboethics: Social and Ethical Implications of Robotics

Gianmarco Veruggio, Fiorella Operto

The present chapter outlines the main social and ethical issues raised by the ever-faster application of robots to our daily life, and especially to sensitive human areas.

Applied to society in numbers and volumes larger than today, robotics is going to trigger widespread social and economic changes, opening new social and ethical problems for which the designers, the end user, the public, and private policy must now be prepared.

Starting from a philosophical and sociological review of the depth and extent of the two lemmas of robotics and robot, this section summarizes the recent facts and issues about the relationship between techno-science and ethics.

The new applied ethics, called roboethics, is presented. It was put forward in 2001/2002, and publicly discussed in 2004 during the First International Symposium on Roboethics.

Some of the issues presented in the chapter are well known to engineers, and less or not known to scholars of humanities, and vice versa. However, because the subject is complex, articulated, and often misrepresented, some of the fundamental concepts relating ethics in science and technology are recalled and clarified.

At the conclusion of the chapter is presented a detailed taxonomy of the most significant ethical legal, and societal issues in Robotics. This study is based on the Euron Roboethics Roadmap, and it is the result of three years of discussions and research by and among roboticists and scholars of Humanities. This taxonomy identifies the most evident/urgent/sensitive ethical problems in the main applicative fields of robotics, leaving deeper analysis to further studies.

64.1	A Methodological Note	1501
64.2	Specificity of Robotics	1502
64.3	What Is a Robot?	1502
64.3.1	Robots Are Nothing Else But Machines.....	1502
64.3.2	Robots (and Technology in General) Have an Ethical Dimension	1502
64.3.3	Robots as Artificial Moral Agents (AMA) ..	1503
64.3.4	Robots: the Evolution of a New Species ..	1503
64.4	Cultural Differences in Robot's Acceptance	1503
64.5	From Literature to Today's Debate	1503
64.6	Roboethics	1504
64.7	Ethics and Morality	1505
64.8	Moral Theories	1505
64.9	Ethics in Science and Technology	1506
64.10	Conditions for Implementation	1507
64.11	Operativeness of the Principles	1507
64.12	Ethical Issues in an ICT Society	1507
64.13	Harmonization of Principles	1509
64.14	Ethics and Professional Responsibility ..	1510
64.15	Roboethics Taxonomy	1511
64.15.1	Humanoids	1511
64.15.2	Artificial Body	1511
64.15.3	Industrial Robotics	1513
64.15.4	Adaptive Robot Servants	1513
64.15.5	Distributed Robotic Systems.....	1514
64.15.6	Outdoor Robotics.....	1514
64.15.7	Surgical Robotics	1515
64.15.8	Biorobotics	1515
64.15.9	Biomechatronics	1515
64.15.10	Health Care and Quality of Life..	1515
64.15.11	Military Robotics	1516
64.15.12	Educational Robot Kits	1517
64.15.13	Robot Toys.....	1518
64.15.14	Entertainment Robotics	1518
64.15.15	Robotic Art	1518
64.16	Conclusions and Further Reading	1519
	References	1522

Many roboticists, as well as authoritative scholars of the history of science and technology, have already labeled the 21st century as the *age of the robots*. Actually, in the course of the present century, intelligent autonomous machines will gradually substitute many automatic machines [64.1].

Humanity has built tools to increase its power by eliminating manual labor and needless drudgery. This factor has become one of the keys to successful economic progress, especially since the Industrial Revolution and the emergence of a mechanized economy, and even more so with the introduction of automatic machines in the 20th century [64.2].

Today, progress in the field of computer science and telecommunications allows us to endow machines with enough intelligence that they can act autonomously. Thus, we can forecast that in the 21st century humanity will coexist with the first *alien* intelligence we have ever come into contact with – robots.

A few years from now, many more fields of application will be robotized, because robotics will have occupied more territories. The figures of the annual *World Robotics Survey*, issued by the United Nations (UN) Economic Commission for Europe and the International Federation of Robotics (IFR), show a steady tendency for growth with the characteristic curve of a rapidly developing field, with short slowdowns and steep climbing.

Certainly, robotics is changing our way of living, working, and operating in the world.

While the application field of robots is widening, the robot is coming out of the factories and into our homes – it is becoming a consumer item. The robot – which was expected to be an extended, intelligent tool for the human – is becoming a partner and a companion [64.3].

Moreover, robotics is also changing our method of conducting scientific inquiry and perhaps even our concept of ourselves [64.4]. Synergies between robotics, neurosciences, medicine, education, and psychology, have broadened the scope of application of the latter, making robotics a platform of global scientific research on humankind, on our galaxy and on the interaction between humankind and nature [64.5].

When robotics is applied to society in numbers and volumes larger than today, it will trigger widespread social and economic changes, for which public and private policy must now be prepared [64.6].

It will be an event rich in ethical, social and economic problems.

In the next decades in the industrialized world – in Japan, South Korea, Europe, United States – humanoid

robots will be among us, companions to the elderly and children, assistants to nurses, physicians, firemen, and workers. They will have eyes, human voices, hands and legs, skin to cover their gears, and brains with multiple functions. Often, they will be smarter and quicker than people they ought to assist. Placing robots in human environments inevitably raises important issues of safety, ethics, and economics.

What is going to happen when these smart robots are our servants and house stewards, and when our lives will depend on them?

Could people who mean no good use these robots to harm others?

The theme of the relationship between humankind and *autonomous* machines appeared early in world literature, developed firstly through legends and myths, then in scientific and moral essays. In early mythology, the ancient peoples expressed their worries about the disrupting power of machines over the old societies: when these artificial creatures to which we have given birth have learned everything from us, or understood that we are weaker than them, will they try to dominate us [64.7]?

In our time, facing the development of ever more powerful computers and the variety of humanoid robots, some scholars and scientists have warned about the dangers of the unlimited use of technology, and especially about the hubristic endeavor to design and manufacture intelligent creatures [64.8,9].

Their concern has been amplified by the harsh discussion around bioengineering and bioethics. The famous physicist and Nobel prize winner Joseph Rotblat said that robotics, genetic engineering, and computer science are threatening the life on our plante [64.8,9].

Thinking computers, robots endowed with artificial intelligence and which can also replicate themselves (...) this uncontrolled self replication is one of the dangers in the new technologies.

Less dramatically, others have pointed out the need to introduce ethical rules in technological applications, especially regarding the behavior of intelligent machines. In this frame, the most matter-of-fact issue is: what will be the cultural and social implications of the robotics invasion? Could robots be dangerous to humankind in any way [64.10]?

Under the pressure of public opinion and the media, roboticists cannot avoid engaging in a critical analysis of the social implications of their researches, in order

to be able to give *scientific and technical*, as well as *philosophical*, answers to questions such as:

- How far can we go in embodying ethics in a robot?
- What kind of *ethics* is robotic ethics?
- How contradictory is, on one side, the need to implement in robots an ethics, and, on the other, the development of robot autonomy?
- Is it right to talk about the *consciousness*, *emotions*, and *personality* of robots [64.11, 12]?

64.1 A Methodological Note

This chapter is by its nature somewhat different from – although complementary to – the remainder of this Handbook, because it deals not only with the scientific and technological issues inherent in the matter, but also with cultural and moral topics related to the introduction of robots in sensitive human areas.

The authors worked on the assumption that:

- This handbook – and in particular the present chapter – is going to be read by roboticists, and also by nonroboticists, by students of robotics as well as by students and scholars of ethics, philosophy of science, sociology, laws, etc. Some of the issues presented here are well known to some, and less or not known to the others, and vice versa. Nonetheless, the authors deemed it useful and important to recall and clarify some of the fundamental concepts relating ethics in science and technology, because the subject is complex, articulated, and often misrepresented.
- Roboethics is an applied ethics that refers to studies and works done in the field of science and ethics (science studies, science and technology studies (S&TS), science technology and public policy, professional applied ethics), and whose main premises are derived from these studies. In fact, roboethics was not born without parents, but it derives its principles from the global guidelines of the universally adopted applied ethics [64.13]. This is the reason why the substantial part of this Chapter is devoted to this subject, before specifically discussing the sensitive areas of roboethics.
- Many of the issues of roboethics are already covered by applied ethics such as computer ethics or bioethics [64.14]. For instance, problems arising in roboethics of dependability, technological addiction, the digital divide, the preservation of human identity and integrity [64.15]; the applications of precautionary principles, economic and social discrimination, artificial system autonomy and accountability, related to responsibilities for (possibly unintended) warfare applications [64.16]; and the nature and impact of human–machine cognitive and affective

bonds on individuals and society have already been matters of investigation in the fields of computer ethics and bioethics [64.16].

The specificity of robotics is underlined from a general point of view. Subsequently, in the taxonomy herein, the specific ethical issues related solely to robotics are carefully evaluated. The present taxonomy is not developed on the basis of affinity to the techno-scientific or disciplinary areas – like the index of the present book. Rather, the roboethics taxonomy is based on the application areas of robots, and on the specificity inherent to the human–robot interaction of these applications [64.17].

In terms of scope, we have taken into consideration – from the point of view of the ethical issues connected to robotics – a temporal range of two decades, in whose frame we could reasonably locate and infer – on the basis of the current state-of-the-art in robotics – certain foreseeable developments in the field.

For this reason, we consider premature – and have only hinted at – problems related to the possible emergence of human qualities in robots: consciousness, free will, self-consciousness, sense of dignity, emotions, and so on. Consequently, this is why we have not examined problems – debated in some other papers and essays – like the proposal to not behave with robots like with slaves, or the need to guarantee them the same respect, rights, and dignity we owe to human workers.

Likewise, and for the same reasons, the target of roboethics is not the robot and its artificial ethics, but the human ethics of the robots' designers, manufacturers, and users.

Although informed about the issues presented in some papers on the need and possibility to attribute moral values to robots' decisions [64.18], and about the chance that in the future robots might be moral entities like – if not more so than – human beings [64.19], the authors have chosen to examine the ethical issues of the human beings involved in the design, manufacturing, and use of the robots.

The authors felt that problems such as those connected with the application of robotics within the

military and the possible use of military robots against some populations not provided with this sophisticated technology, as well as problems of terrorism in robotics and problems connected with biorobotics, implantations, and augmentation, were pressing and serious

enough to deserve a focused and tailor-made investigation. It is absolutely clear that, without a deep rooting of roboethics in society, the premises for the implementation of an artificial ethics in the robots' control systems will be missing.

64.2 Specificity of Robotics

Robotics is a discipline originating from:

- Mechanics
- Automation
- Electronics
- Computer science
- Cybernetics
- Artificial intelligence

but it draws on from several other disciplines:

- Physics/mathematics
- Logic/linguistics
- Neuroscience/psychology
- Biology/physiology
- Anthropology/philosophy
- Art/industrial design

Is robotics a new science? On one side, robotics could be regarded only as a branch of engineering dealing with intelligent, autonomous machines. In this case, it shares experiences with other disciplines, and it is somehow the linear sum of all the knowledge.

On the other side, it could be seen as a new science, in its early stage. Actually, it is the first time in history that humanity is approaching the challenge of replicating a biological organism in the form of an intelligent and autonomous entity. This extraordinary mission gives to robotics the special feature of being a platform where sciences and humanities are converging – an experiment in itself [64.20].

It is not without some grounds that we could forecast that robotics will emerge as a new science, with its own theory, principles, theorems, proofs, and mathematical language [64.21].

However, even before that, robotics displays a specificity, which compels the scientific community to examine closely many of the notions until now applied only to human beings.

Although the authors consider it premature to study scientifically the possible emergence in the robot of human functions, we do not exclude that in the future we will be confronted with problems that today we can only imagine through the work of the artists of the science fiction [64.22, 23].

64.3 What Is a Robot?

From the point of view of how today's society sees robots, we can say that robotics scientists, researchers, and the general public have different evaluations about robots, as described below.

social and ethical implications of robotics fall into the categories of applying ethics to engineering.

64.3.1 Robots Are Nothing Else But Machines

Many consider robots as mere machines: very sophisticated and helpful ones, but always machines. According to this view, robots do not have any hierarchically higher characteristics, nor will the designer provide them with human/animal qualities. In this frame, the issues of the

64.3.2 Robots (and Technology in General) Have an Ethical Dimension

This derives from a conception according to which technology *is not an addition to man but is, in fact, one of the ways in which mankind distinguishes itself from animals*. So that, as language, and computers, but even more so, humanoids robots are symbolic devices designed by humanity to improve its capacity of reproducing itself, and to act with charity and good. “The humanoid (...) is the most sophisticated thinking machine able to assist

human beings in manifesting themselves, and this is ethically very good, as it supposes a radical increment of human symbolic capacity; humanoids will develop a lot of activities in order to increase the human quality of life and human intersubjectivity” [64.24].

64.3.3 Robots as Artificial Moral Agents (AMA)

According to this concept, robots and artificial agents extend the class of entities that can be involved in moral situations, for they can be conceived as moral patients (as entities that can be acted upon for good or evil) and also as moral agents [64.25] (not necessarily exhibiting free will, mental states or responsibility, but as entities that can perform actions, again for good or evil) [64.13].

64.3.4 Robots: the Evolution of a New Species

In the United States, one of the main discussions in the field of ethics and robotics is how to consider robots, as only *objects* or *subjects* which deserve legal rights: robots, not slaves.

According to this point of view, not only will our robotics machines have autonomy and consciousness, emotions and free will, but also humanity will create machines that “*exceed us in the moral as well as the intellectual dimensions*”. Robots, with their rational mind and unshaken morality, will be the new species: Our machines will be better than us, and we will be better for having created them” [64.26].

64.4 Cultural Differences in Robot's Acceptance

While we analyze the present and future role of robots in our societies, we shall be aware of the underlying principles and paradigms that influence social groups and individuals in their relationships with intelligent machines.

Different cultures and religions regard differently intervention in sensitive fields such as human reproduction, neural therapies, implantations, and privacy. These differences originate from the cultural specificities towards the fundamental values regarding human life and death.

In different cultures, ethnic groups, and religions the very concept of *life* and *human life* differ, first of all concerning the immanence or transcendence of human life. While in some cultures women and children have fewer rights than adult males (not even *habeas corpus*), in others the ethical debate ranges from the development to a post-human status, to the rights of robots. Thus, the

different approach in roboethics concerning the rights in diversity (gender, ethnicity, minorities), and the definition of human freedom and animal welfare. From these concepts derive all the other ethical specificities such as privacy, and the border between privacy and traceability of actions.

Cultural differences also emerge in the realm of *natural* versus *artificial*: think of the attitude of different peoples towards surgical or organ implantation. How could human enhancement be viewed [64.27]?

Bioethics has opened important discussions: How is the integrity of the person conceived? What is the perception of a human being?

Last but not least, the very concept of *intelligence*, human and artificial, is subject to different interpretation. In the field of AI and robotics alone, there is a terrain of dispute – let us imagine how harsh it could be outside of the circle of the inner experts [64.4].

64.5 From Literature to Today's Debate

Literature is the instrument by which society expresses itself, free from rigid constraints, and by which it can *simulate* future social developments. Sometimes, by way of literature, important and foresighted scientific issues have been anticipated.

The topic of the threat posed by artificial entities designed by human's ingenuity (legends like *the rebellions of automata*, Frankenstein's myth, the Golem) recurs in classical European literature, as well as the misuse or

the evil use of the product of engineering (the myth of Dedalus). This is not the case in all world cultures. For instance, the Japanese culture does not include such a paradigm; on the contrary machines (and, in general, human products) are always beneficial and friendly to humanity.

In 1942, the outstanding novelist Isaac Asimov, who coined the word *robotics*, formulated his famous three laws of robotics in his novel *Runaround*:

- Law 1: A robot may not injure a human being, or through inaction, allow a human being to come to harm.
- Law 2: A robot must obey the orders given it by human beings except where such orders would conflict with the first law.
- Law 3: A robot must protect its own existence as long as such protection does not conflict with the first or second law.

Later on, in 1983, Asimov added the fourth law (known as the zeroth Law).

- Law 0: No robot may harm humanity or, through inaction, allow humanity to come to harm [64.28, 29].

Although farsighted and forewarning, could these laws really become the *ethics of robots* or are they too *naïve* to be considered seriously in this debate?

64.6 Roboethics

In 2002 the roboticist Gianmarco Veruggio, in the framework of the cultural and educational activity of the Association School of Robotics, started to discuss the need for an ethics which could inspire the work of robotics scientists. He called this new applied ethics, roboethics.

Roboethics is an applied ethics whose objective is to develop scientific/cultural/technical tools that can be shared by different social groups and beliefs. These tools aim to promote and encourage the development of Robotics for the advancement of human society and individuals, and to help preventing its misuse against humankind [64.31].

According to the definition, roboethics is not the *ethics of robots*, nor any artificial ethics, but it is the human ethics of robots' designers, manufacturers, and users.

In January 2004, in Sanremo, Italy, the authors, in collaboration with roboticists and philosophers, organized the First International Symposium on Roboethics, where the word roboethics was officially used for the first time.

On this occasion Paolo Dario (RAS president 2002-03) and Kazuo Tanie (RAS president 2004-

Over the last few decades, scientific and technological developments have brought forward the frontiers of robotics, so that those problems that years ago seemed only theoretical, or a matter of literature and science fiction, are becoming very practical, and even urgent.

Some of these problems have alerted the robotics community on the need to open a discussion on the principles that should inspire the design, manufacturing, and use of robots.

In 2001, the collaboration between the roboticist Paolo Dario and the philosopher José Maria Galván expressed the concept of technoethics [64.30].

In the same year, on the occasion of Italy–Japan 2001 (Tokyo, Japan), Paolo Dario and Japanese roboticist Atsuo Takanishi organized the Workshop *Humanoids. A Techno-Ontological Approach*, which was held at Waseda University. The lecture given by Galván was published in the December 2003 issue of *IEEE Robotics & Automation Magazine*, *On Technoethics* [64.24].

05) established a technical committee (TC) on roboethics, with the aims of providing the *IEEE Robotics and Automation Society* with a framework for analyzing the ethical implications of robotics



Fig. 64.1 The Roboethics' logo, sketched by the renowned Italian artist Emanuele Luzzati (1920 – 2007), is represented by a young smiling girl receiving a flower from a chivalrous humanoid robot

research, by promoting the discussion among researchers, philosophers, ethicists, and manufacturers, but also by supporting the establishment of shared tools for managing ethical issues in this context.

In 2005, the European Robotics Research Network (EURON) funded the project called the EURON Roboethics Atelier, with the aim of drawing the first roboethics roadmap. In 2006, in Genoa, Italy, scholars from humanities met for three days with engineers and

roboticists to draw the lines of the EURON roboethics roadmap [64.17].

Roboethics is not a veto or a prohibitionist ethics. Its main lines of development are: the promotion of culture and information; the permanent education; a vigorous and straight public debate; and the involvement in all these activities of the young generations who are the actors of the future [64.32].

Now, it is worth analyzing briefly the general principle of ethics.

64.7 Ethics and Morality

Ethics is the branch of philosophy concerned with the evaluation of human conduct [64.33].

The difference between ethics and morality is subtle. According to Italian philosopher Remo Bodei: “The word Ethics is generally associated to our relationship with others, to our public dimension; while morality concerns more with our conscience’s voice, our relationship with ourselves. The distinction, however, is purely conventional, because the word comes from the Greek word *ethos*, which means habit, and morality from Latin *mos/moris*, which again means habit.”

Another definition is the following:

In simple terms morality is the right or wrong (or otherwise) of an action, a way of life or a decision, while ethics is the study of such standards as we use or propose to judge such things [64.34].

In short *morality* is the subject of a science called *ethics* (although *morality* may also refer to a code of conduct: see <http://plato.stanford.edu/entries/morality-definition/>) [64.35].

64.8 Moral Theories

Apart from *virtue ethics*, the classical Greek moral philosophy, the dominant moral theories are:

- *Utilitarianism* - or more generally *consequentialism*: guideline properties that depend only on the consequences, not on the circumstances or the nature of the act in itself;
- *Contractualism*: morality as the result of an imaginary contract between rational agents, who are agreeing upon rules to govern their subsequent behavior. The idea is not that moral rules have resulted from some explicit contract entered into by human beings in an earlier historical era, a claim that is almost certainly false. (John Locke seems to have held a view of this sort.) Nor is the idea that we are, now, implicitly committed to a contract of the *I will not hit you if you do not hit me* variety, which implausibly reduces moral motivation;

- *Deontologism*, or duty-based ethics. What is my moral duty? What are my moral obligations? How do I weigh one moral duty against another? Kant’s theory is an example of a deontological or duty-based ethics: it judges morality by examining the nature of actions and the will of agents rather than the goals achieved.

In scientific circles, *secular humanism* – a nontheistically ethical philosophy based upon naturalism, rationalism, and free thought – has gained great importance and influence [64.36].

It is true that in the scientific and technological domain a professional conception of ethics, closer to professional deontology, is becoming dominant and a universal standard of practice.

Furthermore, ethics in the digital world needs new approaches, beyond the classical moral theories, opening new and unresolved moral problems.

64.9 Ethics in Science and Technology

In the last years, concerned scientists, stakeholders, nongovernmental organizations (NGOs), parents, and consumers associations have increased their influence on the development of the scientific and technological researches, proposing (often imposing) to scientists, manufacturers, distributors, and advertising agencies the adoption of ethical conducts. Sometimes their intervention was mild, other times it had the result of closing down wealthy lines of research.

That is one of the reasons why we cannot underestimate the impact of society's opinions on *science and society* issues, and on the trend of the advancement of science and technology.

How can ethical concerns and visions become practical rules of society [64.37]? How can the ethical principles discussed in transdisciplinary assemblies; expressed by warnings or the public's concern; suggested by religious personalities, theologians, and moral leaders; and/or forwarded by a community of concerned scientists modify research and development (R&D) [64.38]? How can ethical thrust be embodied in the R&D activity without imposing on it unjustified restrictions, so depriving the scientist of his/her own freedom of thought [64.39]?

Through the millennia of the history of science and technology, society has envisaged ways to express their ethical concern [64.40].

The professional *oath* is either a statement or a promise expressed by a new entry into professional careers to be faithful to the traditional values of the professional order he/she is entering in. The ancient *Hippocratic oath* is the recurrent example for other initiatives to develop and implement codes of conduct for scientists in general, and in specific areas in particular.

Otherwise, a *manifesto* is a public declaration of intentions, opinions, objectives or motives, often issued by a private organization or a government. For example, the Russell–Einstein Manifesto of 1955 is a public declaration against war and the further development of weapons of mass destruction.

A *statement* or a *declaration* can be employed to underline a given topic. As such, it can be either weakly or strongly prescriptive, morally or legally binding.

During the World Robot Conference which took place in Fukuoka, Japan, the participants released a three-part list of *expectations for next-generation robots*, called the *World Robot Declaration* issued on 25 February 2004. It states that:

- Next-generation robots will be partners that coexist with human beings;
- Next-generation robots will assist human beings both physically and psychologically;
- Next-generation robots will contribute to the realization of a safe and peaceful society.

A *recommendation* serves to induce acceptance or favor. It is a prescription only in the weak sense of offering advice: a normative suggestion that is neither legally nor morally binding. More conclusive is the *appeal*, an earnest request for support: a petition, entreaty, or plea.

A *resolution* is a formal expression of opinion or intention made (usually after voting) by a formal organization, legislature, or other group.

In the last 50 years, many professional associations have adopted their *code*: a written text that offers a collection of laws, regulations, guidelines, rules, directives or principles for moral conduct.

The *guiding principles* of the Code of Research Ethics are *non-maleficance* and *beneficence*, indicating a systematic regard for the rights and interests of others in the full range of academic relationships and activities. *Non-maleficance* is the principle of doing, or permitting, no official misconduct. It is the principle of doing no harm in the widest sense. *Beneficence* is the requirement to serve the interests and well being of others, including respect for their rights. It is the principle of doing well in the widest sense.

In the field of roboethics, the Government of Japan through the Ministry of Economy, Trade, and Industry has issued a *hugely complex set of proposals*, which is an articulated set of guidelines to ensure a safe deployment of robots in nonstructured environments. Under these guidelines, all robots would be required to report back to a central database any and all injuries they cause to the people they are meant to be helping or protecting. The draft is currently open to public comment with a final set of principles being made public in 2007. Among the indications:

Via a structure of general regulation and the adoption of that regulation, the planning, manufacturing, administration, repair, sales and use of robots shall observe the need for safety at every stage (...) The reasonably predictable misuse of robots shall be defined as the management, sale and use of next-generation robots for purposes not intended by manufacturers (...) There should, in principle, be no serious accidents such as fatal accidents involving

robots, and the frequency of such accidents should be lowered as far as possible. Affordable multiple security measures should be taken in case one protection method alone is insufficient.

The *charter* is an ancient form of agreement. An example is the charter of the United Nations. Charters have a legal character and are connected, in principle, to sanctions when not properly executed.

In 2007, the Government of the Republic of Korea announced the birth of a governmentally sponsored working group whose aim is the definition of a roboethics charter.

The process towards the Korean Roboethics Charter is the following. The first step concerns the establishing of a working group (WG) on roboethics composed by robot developers, chief executive officers (CEOs), psychologists, futurists, writers, government officials, users, lawyers, and doctors. The WG will release a draft

that will be circulated for feedbacks among online international communities, and through public hearings [64.41].

The revised draft will go for deliberation to the *Robot Industry Policy Forum*, which will be composed of 40 members, representing the main stakeholders. Subsequently, the draft will go to the *Industrial Development Council* (composed of 29 members). At this point – presumably at the end of 2007 – the draft becomes the Korean Roboethics Charter, and it will be officially announced. Then, application rules and detailed guideline will be released.

Other means of implementing ethical concerns in science and technology are the *convention*, a form of agreement, or a contract, and also a practice established by general consent.

Then, principles established by a government applicable to a people and enforced by judicial decision become *law*.

64.10 Conditions for Implementation

Once the chosen code of research ethics has been defined, a list of conditions for implementation should be drawn up. Actually, no regulation can be implemented without at least some of those conditions, which should favor the application of the rules.

From the *individual* scientist's point of view, he/she has to guarantee some conditions, without which he/she is not in the position to adhere to nor to implement the Code of Ethics. These are: decision-making capacity, that is the empowered position and freedom to identify and choose alternatives based on the values and preferences defined and accepted; individual scientists' honesty and integrity; and transparency of processes.

On the other side, the given scientific institution, and in the final analysis society, should guarantee the individual scientist the reasonable general framework in which he/she finds the best conditions to work. These are:

- Periodic review of the application procedures
- Review and assistance by ethics committees
- Promotion of public debate
- Definition of risk assessment, management and prevention
- Transnational practices: comparison of conducts across countries and comparisons of professional ethics around the world

64.11 Operativeness of the Principles

The implementation of regulations or of codes of conduct should provide guidelines for operationalizing and reconciling the principles to be implemented, in case such principles appear inherently contradictory.

For instance, ethical guidelines may – by virtue of their collective nature – pose a threat to the individual's moral autonomy. Or, the public's demand for accountability could threaten the professions' pursuit of autonomy.

64.12 Ethical Issues in an ICT Society

The importance of ethics in science and technology has been demonstrated by our recent history. Three of

the front-rank fields of science and technology: nuclear physics, bioengineering, and computer science, have al-

ready been forced to face the consequences of their research's applications because of pressure caused by dramatic events, or because of the concern of the general public.

The introduction of intelligent machines in our daily life brings up global social and ethical problems which are usually summarized as

- dual-use technology (every technology can be used and misused)
- anthropomorphization of technological products (it is well known and documented that people attribute intentions, goals, emotions, and personalities to even the simplest of machines with life-like movement or form)
- humanization of the human-machine relationship (cognitive and affective bonds toward machines)
- technology addiction;
- digital divide, socio-technological gap (per ages, social layer, per world areas)
- fair access to technological resources
- the effects of technology on the global distribution of wealth and power
- the environmental impact of technology

Due to the interdisciplinarity of robotics, roboethics shares problems and solutions with other applied ethics: computer ethics, information ethics, bioethics, technoethics and neuroethics.

Computer ethics (CE), a term coined by Walter Maner in the mid 1970s, denotes the field of research that studies ethical problems *aggravated, transformed or created by computer technology*.

Perhaps the first contact between ethics and computer science took place in the 1940s, when Norbert Wiener, professor at the MIT and one of the founding fathers of computer science, expressed his concern about the social effects of the technologies he himself contributed to develop [64.42]. In 1948, in his book *Cybernetics: or Control and Communication in the Animal and the Machine*, and in his following book, *The Human Use of Human Beings*, he pointed out the dangers of nuclear war and the role of scientists in weapons development in 1947, shortly after Hiroshima. Although he did not use the term *computer ethics* he laid down a comprehensive foundation for computer ethics research and analysis. Wiener's foundation of computer ethics was far ahead of its time [64.43, 44].

It was not until 1968 that Wiener's concern became actual practice, when Donn Parker, one of the most famous scientist of the Stanford Research Institute (SRI) at Menlo Park, began to examine unethical and ille-

gal uses of computers by computer professionals. He writes:

It seemed that when people entered the computer center, they left their ethics at the door.

In 1968 he published his *Rules of Ethics in Information Processing* and promoted the development of the first code of professional conduct of the Association for Computing Machinery (ASM), which was adopted by the ACM in 1973.

During the late 1960s, Joseph Weizenbaum, the designer of the computer program Eliza, shocked by the emotional involvement of psychiatric scholars towards his simple programs, expressed his concern that an *information processing model* of human beings was reinforcing an already growing tendency among scientists, and even among the general public, to see humans as mere machines. Weizenbaum wrote the book *Computer Power and Human Reason*, in which he expressed his thoughtful ethical philosophy [64.45].

In the late 1970s, Walter Maner of the Virginia Old Dominion University was the first to employ the label *computer ethics* to define the field of inquiry dealing with ethical problems aggravated, transformed, or created by computer technology [64.46].

In 1985, James Moor of Dartmouth College published his article *What is Computer Ethics?* [64.47], and Deborah Johnson of the Rensselaer Polytechnic Institute published her book, *Computer Ethics*, the first textbook – and for more than a decade, the defining textbook – in the field. In 1983 the Computer Professional for Social Responsibility (CPSR) was founded at Palo Alto: a global organization promoting the responsible use of computer technology. Incorporated in 1983 (following discussions and organizing that began in 1981), CPSR is the first international association whose mission is to educate policymakers and the public on a wide range of issues [64.48].

In 1991 computer ethics was officially added as a subject to the programs in the computer science departments of the United States.

In the 1990s, it was proposed that the core of the issues of CE did not lie in the specific technology, but in the raw material manipulated by it (data/information), as a result of which several researchers (especially the team at Oxford led by Luciano Floridi) developed *information ethics* (IE).

Bioethics is the study of the ethical, social, legal, philosophical, and other related issues arising in health care and in the biological sciences (International Association of Bioethics, IAB) [64.49, 50].

In 1970 Van Rensselaer Potter (1911–2001) coined the term *bioethics* [64.50]. He was an American biochemist, Professor of Oncology at the McArdle Laboratory for Cancer Research at the University of Wisconsin, Madison. The first appearance of the term was in his book *Bioethics, A Bridge to the Future*. He coined it after trying for many months to find the right words to express the need to balance the scientific orientation of medicine with human values.

Potter's original concept of bioethics was comprised of a global integration of biology and values designed to guide human survival, with a new bioethics as the bridge between science and humanities. Increasingly, he felt the need to link what he came to realize had become mainstream biomedical ethics with environmental ethics.

During his career he continued to modify the term bioethics to differentiate his conceptions from the dominant view of biomedical ethics. He eventually selected the term global bioethics and this became the title of his second book [64.51]. In it, there is a new definition of the term bioethics, as *biology combined with diverse humanistic knowledge forging a science that sets a system of medical and environmental priorities for acceptable survival*.

The field of bioethics is at a critical stage of evolution, having now passed the 13th year of the development of bioethics programs. It is in a phase of professionalization attending to both the ethical framework for clinical and industrial bioethical consultation and the creation of the next level of academic organizational success, namely departments and PhD programs [64.52].

Technoethics is a recent definition, derived from Christian theology,

as a sum total of ideas that bring into evidence a system of ethical reference that justifies that profound dimension of technology as a central element in the attainment of a finalized perfection of man [64.53].

Neuroethics is concerned with the ethical, legal, and social policy implications of neuroscience, and with aspects of neuroscience research itself [64.54]. Neuroethics encompasses a wide array of ethical issues emerging from different branches of clinical neuroscience (neurology, psychiatry, psychopharmacology) and basic neuroscience (cognitive neuroscience, affective neuroscience).

64.13 Harmonization of Principles

Internationally recognized institutions such as the United Nations, the World Health Organization (WHO), the Food and Agricultural Organization (FAO), the UN Educational, Scientific, and Cultural Organization (UNESCO)'s World Commission on the Ethics of Scientific Knowledge and Technology (COMEST), the International Labor Organization (ILO), the World Medical Association, the World Summit on the Information Society, and the European Union have identified general ethical principles that have been adopted by most nations, cultures, and people of the world.

Furthermore, the international scientific, juridical, economic, and regulatory community has on many occasions proposed a harmonization of world ethical principles applied to science and technology, especially in those cases when these principles involve sensitive issues such as life, human reproduction, human dignity, and freedom.

The Ethics of Science and Technology Programme, part of UNESCO's Division of Ethics of Science and Technology in the Social and Human Sciences Sector, and COMEST, an advisory body to UNESCO composed

of 18 independent experts, have proposed, in the field of bioethics, to start a process towards a *declaration on universal norms on bioethics*. In Rio de Janeiro in December 2003, COMEST organized an international conference on the issue of a *universal ethical oath for scientists*.

In Europe, the 6th Framework Program, funded, under the Science and Society work Programme, the ETHICBOTS project (an abbreviation for merging technoethics of human interaction with communication, bionic, and robotic systems). The project aims to promote and coordinate a multidisciplinary group of researchers in artificial intelligence, robotics, anthropology, moral philosophy, philosophy of science, psychology, and cognitive science, with the common purpose of identifying and analyzing technoethical issues concerning the integration of human beings and artificial (software/hardware) entities. Three kinds of integration are analyzed:

1. Human–softbot integration, as achieved by AI research on information and communication technologies

2. Human–robot noninvasive integration, as achieved by robotic research on autonomous systems inhabiting human environments
3. Physical, invasive integration, as achieved by bionic research

64.14 Ethics and Professional Responsibility

Although ethics in science and technology is not limited to deontology or professional ethics, but concerns a broader range of questions involving the fundamental beliefs and moral principles, its results and conclusions become guidelines for conduct in professional daily life.

From the social and ethical standpoints, in deciding the design, development, and application of a new technology, designers, manufacturers, and end users should be following rules, which are common to all human beings:

- human dignity and human rights
- equality, justice, and equity
- benefit and harm
- respect for cultural diversity and pluralism
- nondiscrimination and nonstigmatization
- autonomy and individual responsibility
- informed consent
- privacy and confidentiality
- solidarity and cooperation
- social responsibility
- sharing of benefits
- responsibility towards the biosphere
- obligatory cost-benefit analysis (whether ethical issues are to be considered as part of a proper cost-benefit analysis)
- exploiting potential for public discussion

(the Charter of Fundamental Rights of the European Union, 2001 [64.55]).

Computer and information ethics has developed a codes of ethics called *PAPA* (an acronym of: privacy, accuracy, intellectual property, and access), which could be adopted by robotics. It is composed as follows.

- *Privacy*: What information about ones self or ones associations must a person reveal to others, under what conditions, and with what safeguards? What things can people keep to themselves and not be forced to reveal to others?
- *Accuracy*: Who is responsible for the authenticity, fidelity, and accuracy of information? Similarly, who

is to be held accountable for errors in information and how is the injured party to be made whole?

- *Property*: Who owns information? What are the just and fair prices for its exchange? Who owns the channels, especially the airways, through which information is transmitted? How should access to this scarce resource be allocated?
- *Accessibility*: What information does a person or an organization have a right or a privilege to obtain, under what conditions, and with what safeguards?

Problems of the *delegation* and *accountability* to and within technology are problems of daily life for every one of us. Today, we give responsibility for crucial aspects of our security, health, life-saving, and so on to machines.

Professionals are advised to apply, in performing sensitive technologies, the *precautionary principle*:

When an activity raises threats of harm to human health or the environment, precautionary measures should be taken even if some cause-and-effect relationships are not fully established scientifically.

(Source: January 1998 Wingspread Statement on the Precautionary Principle; see also the Rio Declaration from the 1992 United Nations Conference on Environment and Development, Agenda 21; and the Commission of the European Communities, Brussels, 02.02.2000, com (2000) 1 communication from the Commission on the precautionary principle.)

From the precautionary principle other rules can be derived, such as:

- noninstrumentalization
- nondiscrimination
- Informed consent and equity
- Sense of reciprocity
- Data protection

All over the world, associations and orders of engineers have adopted codes of ethics guiding towards responsible conduct in research and practice. In this context,

security and *reliability* are the most important ethical codes of conduct.

Among the other important recommendations are the following:

- Hold paramount the safety, health, and welfare of the public in the performance of their professional duties.
- Perform services only in areas of their competence.

- Issue public statements only in an objective and truthful manner.
- Act in professional matters for each client as faithful agents or trustees.
- Avoid improper solicitation of professional assignments.

(From the *American Council of Engineering Companies Ethical Guidelines*).

64.15 Roboethics Taxonomy

In this section we outline a first classification of the most evident ethical issues of robotics, based on the EURON roboethics roadmap.

Certainly, classifying the different branches of robotics is not an easy task. Likewise, it is a complex undertaking to organize a matrix of field of robotics/ethical issues. We have tried to classify these topics according to homogeneous fields from an applicative point of view.

Furthermore, in the present taxonomy, we have chosen the *triage process* of identify the most evident/urgent/sensitive ethical problems in robotics, leaving to other times and further studies more complex problems.

64.15.1 Humanoids

One of the most ambitious aims of robotics is to design an autonomous robot that could reach – and even surpass – human intelligence and performance in partially unknown, changing, and unpredictable environments.

Artificial intelligence will be able to lead the robot to fulfill the missions required by the end users. To achieve this goal, over the past decades scientists have worked on AI techniques in many fields, including:

1. Artificial vision
2. Perception and analysis of the environment
3. Natural language processing
4. Human interaction
5. Cognitive systems
6. Machine learning and behaviors
7. Neural networks

In this context, one of the fundamental aspects of the robots is their capability to learn: to learn the characteristics of the surrounding environment, that is, (1) the physical environment, but also (2) the living beings that inhabit it. This means that robots working in a given en-

vironment have to distinguish human beings from other *objects*.

In addition to learning about their environment, robots have to learn about their own behavior, through a self-reflective process. They have to learn from the experience, replicating somehow the natural processes of the evolution of intelligence in living beings (synthesis procedures, trying-and-error, learning by doing, and so on).

It is almost inevitable that human designers are inclined to replicate their own conception of intelligence in the intelligence of robots. In turn, the former gets wired into the control algorithm of the robots. Robotic intelligence is a learned intelligence, fed by the world models uploaded by the designers. It is a self-developed intelligence, evolved through the experience which robots have gained through the learned effects of their actions. Robotic intelligence also includes the ability to evaluate and attribute a judgment to the actions carried out by robots.

All these processes embodied in the robots produce a kind of intelligent machine endowed with the capability to express a certain degree of autonomy. It follows that a robot can behave, in some situations, in a way that is unpredictable to their human designers. Basically, the increasing autonomy of the robots could give rise to unpredictable and nonpredictable behaviors.

So, without necessarily imagining some science-fiction scenarios where robots are provided with consciousness, free will, and emotions, in a few years we are going to be cohabiting with robots endowed with self-knowledge and autonomy – in the engineering meaning of these words.

64.15.2 Artificial Body

Humanoids are robots whose body structure resembles the human one. They answer an age-old dream of hu-

manity, and certainly do not spring only from rational, engineering, or utilitarian motivations, but also from psychoanthropological ones.

Humanoids are the expression of one of the demands of our European culture, that is, that humankind be the creator of some mechanical being in the shape of a human. In Japanese culture, it is the demand to carefully replicate nature in all its forms.

This is a very difficult and demanding enterprise, a project of the level of the mission to the moon. However, precisely because it is one of humanity's dreams, large investments are being made and progress is quick.

It has been forecast that in the not-so-distant future we will cohabit with humanoids whose shape will be so similar to that of human beings that it will render it possible to get mixed up in certain situations with the latter. Humanoids will assist human operators in human environments, will replace human beings, and will cooperate with human beings in many ways.

Given the high cost and the delicacy of the humanoids, they will probably be employed in tasks and in environments where the human shape would really be needed, that is, in all these situations where the human-robot interaction is primary, compared to any other mission – human-robot interactions in health care; children/disable people/elderly assistance; baby sitting; office clerks, museum guides; entertainers, sexual robots, and so on. Or, they will be employed as testimonials for commercial products.

The special tasks humanoid robots can fulfill are manifold. Humanoids are robots so adaptable and flexible that will be rapidly used in many situations and circumstances. They can assist humans to perform very difficult tasks, and behave like true and reliable companions in many ways. Their shape, and the sophisticated human–robot interaction, will be very useful for situations in which a human shape is needed.

The research carried out in humanoids laboratories throughout the world will have as a side-effect the development of a platform to study the human body, for training, haptic testing, and training, with extraordinary results for healthcare, education, edutainment, and so on [64.56].

Faced with an aging population, the Japanese society see humanoids robots as one way to enable people to continue to lead an active and productive life in their old age, without being a burden to other people.

From the point of view of *safety* in the use of humanoids, and taking into account that in the not distant future they will be used as companions to human beings, humanoids can rise serious problems related to the

reliability of their *internal evaluation systems* and to the *unpredictability* of robots' behavior. Thus, designers should guarantee the traceability of evaluation/actions procedures, and the identification of robots.

Concerning safety, it should be underlined that an incorrect action by humanoids can lead to a dangerous situation for living beings and the environment. Furthermore, there could be also the case where the incorrect action by the robot is caused by a criminal intent, if robot's autonomy was controlled by ill-intentioned people, who modified the robot's behavior in a dangerous and fraudulent course.

Because humanoids combine almost all of the characteristics of the whole spectrum of robots, their use implies the emergence of nearly all of the problems we will examine below. In particular, their introduction into human environments, workplaces, homes, schools, hospitals, public places, offices, and so on, will deeply and dramatically modify our society.

There is already an important and well-documented literature on the implication of coexistence between human beings and humanoids. The problems range from the replacement of human beings (economic problems; human unemployment; reliability; dependability; and so on) to psychological problems (deviations in human emotions, problems of attachment, disorganization in children, fears, panic, confusion between the real and the artificial, feeling of subordination towards robots) [64.57].

On the technological and scientific side, trust towards and ever-greater autonomy of humanoids (and of the robots in general) are the dominant trends. From the ethical standpoint, many have expressed fear that too much autonomy can harm human beings. For instance, Japan's Ministry of Economy, Trade, and Industry are working on a new set of safety guidelines for next-generation robots. This set of regulations would constitute a first attempt at a formal version of the first of Asimov's science-fiction *laws of robotics*, or at least the portion that states that humans shall not be harmed by robots

Recently, Japan's ministry guidelines will require manufacturers to install a sufficient number of sensors to prevent robots from running into people. Lighter or softer materials will be preferred, to further prevent injury. Emergency shut-off buttons will also be required.

Another set of questions arises around the shape of the humanoids. Is it right that robots can exhibit a *personality*? Is it right that robot can express *emotion*? The concern expressed by psychologists is that, well before evolving to become conscious agents, hu-

manoids can be an extraordinary tool used to control human beings.

In one of their papers, Wagner, Cannon, Van der Loos [64.58] list the main questions posed by the introduction of a new technology:

- Under what conditions should we decide that deployment is acceptable?
- At what point in the development of the technology is an increase in deployment acceptable?
- How do we weigh the associated risks against the possible benefits?
- What is the rate of the ethics of functional compensation or repair versus enhancement? This issue is especially notable regarding the problem of *augmentation*: In some cases a particular type of technology is regarded as a way of compensating for some function that is lacking compared to the majority of humans; in other cases, the same technology might be considered an enhancement over and above that which the majority of humans have. Are there cases where such enhancement should be considered unethical?
- Are there cases where a particular type of technology itself should be considered unacceptable even though it has the potential for compensation as well as enhancement?
- The question of identifying cause, and assigning responsibility, should some harm result from the deployment of robotic technology [64.59].

64.15.3 Industrial Robotics

An industrial robot is officially defined by ISO as an automatically controlled, reprogrammable, multipurpose manipulator.

Typical applications of industrial robots include welding, painting, ironing, assembly, pick and place, palletizing, product inspection, and testing, all accomplished with high endurance, speed, and precision.

Complexity can vary from simple single robot to very complex multirobot systems:

- Robotic arms
- Robotic work cells
- Assembly lines

From the social and economic standpoint, the benefits of these robots are extraordinary. They can relieve human beings of heavy work, dangerous workplaces, and routine and tedious activities.

In the future, we can imagine robotic factories, completely managed by robots. In the industrialized countries, which are facing a looming labor shortage due to their aging populations, robots in factories will cut costs.

Industrial robots increase productivity (higher speed, better endurance); they increase quality (precision, cleanliness, endurance); they make highly miniaturized devices possible (building the European Robotics Platform, EUROP).

Social problems stemming from the introduction of robots in factories are, first of all, loss of jobs and unemployment. On the other hand, while a welfare policy is to be implemented at a national level to facilitate workers' redeployment, and educational programs to create new skills, it should also be said that robots have also created new jobs directly and can create wealth, leading to the development of new industries and workplaces.

64.15.4 Adaptive Robot Servants

Robots come in several shapes and sizes (wheeled, legged, humanoids), equipped with different kinds of sensing systems (artificial vision systems, ultrasonic, radio) and manipulations (grippers, hands, tools, probes). Service robots support and back up human operators.

According to the UN's annual World Robotics Survey issued by the UN Economic Commission for Europe (UNECE) and the International Federation of Robotics, 607,000 automated domestic helpers were in use at the end of 2003, two-thirds of them purchased during that year. The survey forecasts that the use of robots around the home – to mow lawns, vacuum floors and manage other chores – will increase year on year.

By the end of the decade, the study said, robots will *not only clean our floors, mow our lawns and guard our homes but also assist old and handicapped people with sophisticated interactive equipment, carry out surgery, inspect pipes and sites that are hazardous to people, fight fire and bombs.*

Servant robots can: clean and housekeep; they are fast and accurate, and never bored. They can babysit, because they are patient, talkative, and able to play many games, both intellectual and physical. They can assist patients, the elderly, and the handicapped in clinics or at home, being always available, reliable, and taught to provide physical support.

Certainly, servant robots can guarantee a better quality of life, providing that designers guarantee safety and security (unpredictability of machine behavior from ma-

chine learning; assignment of liability for misbehavior or crime).

From a social and psychological standpoint, overuse could lead to technology addiction or invasion of privacy. Humans in robotized environments could face psychological problems [64.60].

64.15.5 Distributed Robotic Systems

The fast growth of the many wireless systems makes it possible to link all robots to the Web. Network robotics will allow remote human–robot interaction for teleoperation and telepresence, and also robot–robot interaction for data sharing, and cooperative working and learning. When Web speed become comparable to that of the internal local-area network (**LAN**) of the robot, the machine will explode into a set of specialized systems distributed over the net.

Complex robotic systems will be developed, constituted by a team of cooperating robotic agents/components connected through information and communication technology (**ICT**) and **GRID**, on distributed computing, technologies:

- Networked knowledge system
- Networked intelligence systems
- Multirobot systems

Multirobot systems are self-organizing robot teams consisting of a large number of heterogeneous team members. The organization in robot teams or squads is needed to perform specific tasks that require automatic task distribution and coordination at a global and local level, and when central control becomes impossible due to large distances and the lack of local information, or when signal transmission delays.

A full-scale robot team would be of tremendous value in a number of applications such as security, surveillance, monitoring, gardening, and pharmaceutical manufacturing. In addition, the coordination of heterogeneous teams of robots will also be of significant value in terms of planning, coordination, and the use of advanced manufacturing systems.

The benefits of robot teams are manifold, including increases in efficiency in performing complex tasks, and the capability to manage large-scale applications. They also provide abundant and replaceable interchangeable agents, which improves the reliability because the group can perform even after losing most of its parts.

On the other side, scientists should be aware of some of the risks in applying robot teams, for instance, the increasing dependability of primary services from com-

plex systems, and the unpredictability of robot team behavior. From a criminal point of view the assignment of liability for misbehavior or crimes, vulnerability to hacking, and concerns about privacy are some of the important issues.

64.15.6 Outdoor Robotics

Outdoor robots are intelligent machines that explore, develop, secure, and feed our world. Robots could also be employed in dangerous operations such as laying explosives, going underground after blasting to stabilize a mine roof, and mining in areas where it is impossible for humans to work or even survive.

They can work in the following environments:

Land

- Mining (automated load–haul–dump trucks, robotic drilling and blasting devices)
- Cargo handling (cranes and other automation technology for cargo lift on/lift off)
- Agricultural (autonomous tractors, planters and harvesters, applicators for fertilizers and pest control)
- Road vehicles (autonomous vehicles for humans or cargo transportation)
- Rescue robotics (robots that support first-response units in disaster missions)
- Humanitarian demining (robots for detecting, localizing, and neutralizing landmines)
- Environmental protection (robots for pollution cleaning and decommissioning of dangerous facilities).

Sea

- Research (marine robots for oceanography, marine biology, geology)
- Offshore (underwater robots for inspection, maintenance, repair and monitoring of oil and gas facilities in deep and ultradeep waters)
- Search and rescue (underwater robots for first-response intervention in case of accidents at sea, such as a submarine that has run aground).

Air

- **UAV** (autonomous airplanes for weather forecasting, environmental monitoring, road traffic control, large-area survey, and patrolling).

Space

- Space exploration (deep-space vehicles, landing modules, rovers)

- Space stations (autonomous laboratories, control and communication facilities)
- Remote operation (autonomous or supervised dexterous arms and manipulators)

Mobile robots in particular can be highly valuable tools in urban rescue missions after catastrophes such as earthquakes, bomb or gas explosions, or everyday incidents such as fires and road accidents involving hazardous materials. Robots can be used to inspect collapsed structures, to assess the situation and to search and locate victims.

Among the benefits of employing such robots is the increased efficiency of the exploitation of natural resources, which could increase food production for the world's population.

Concerning space robotics, it is obvious that, on the basis of current knowledge and technology, the robot can be our pioneer in space travel and missions to explore the far planets of the solar system and beyond.

On the social front, the unrestrained use of outdoor robots could extend the excessive anthropization and exploitation of the planet, which can become in turn a threat to biodiversity and all other forms of life on the planet. As for AI, the other branch of robotics, this could lead to technology addiction. Furthermore, given the versatility of these robots, they can be converted from civilian use for warfare and misuse (terrorism, pollution).

64.15.7 Surgical Robotics

The field of surgery is entering a time of great change, spurred on by remarkable recent advances in surgical and computer technology. Computer-controlled diagnostic instruments have been used in the operating theater for years to help provide vital information through ultrasound, computer-aided tomography (CAT), and other imaging technologies. Recently robotic systems have made their way into the operating room as dexterity-enhancing surgical assistants and surgical planners, in answer to surgeons' demands for ways to overcome the surgical limitations of minimally invasive laparoscopic surgery, a technique developed in the 1980s. On 11 July 2000, the Food and Drug Administration (FDA) approved the first completely robotic surgical device.

Typical applications are:

- Robotic telesurgical workstations
- Robotic devices for endoluminal surgery
- Robotic systems for diagnosis (Cat Scan - Computerized Axial Tomography Scan; NMR, Nuclear

magnetic resonance; PET - Positron emission tomography)

- Robots for therapy (laser eye treatment, targeted nuclear therapy, ultrasonic surgery, etc.)
- Virtual environments for surgical training and augmentation
- Haptic interfaces for surgery/physiotherapy training

64.15.8 Biorobotics

Biorobotics comprises many different but integrated field of researches. Among them, the design and fabrication of novel, high performance bio-inspired machines and systems, for many different potential applications. The development of nano/ micro/ macro devices that can better act on, substitute parts of, and assist human beings - in diagnosis, surgery, prosthetics, rehabilitation and personal assistance. The development of devices for biomedical applications (e.g. mini-invasive surgery and neuro-rehabilitation).

Biorobotics is a new scientific and technological area with a unique interdisciplinary character. It derives its methodology mainly from the sectors of robotics and biomedical engineering, but also includes knowledge from, and provides useful applications to, many sectors of engineering, basic and applied sciences (medicine, neuroscience, economics, law, bio/nanotechnologies in particular), and even the humanities (philosophy, psychology, ethics).

Biorobotics offers a new paradigm for engineers. The engineer no longer just cooperates with neuroscientists, but has also become a scientist in order to discover basic biological principles that make their job easier [64.61].

64.15.9 Biomechatronics

Human prostheses for locomotion, manipulation, vision, sensing, and other functions include:

- Artificial limbs (legs, arms)
- Artificial internal organs (heart, kidney)
- Artificial senses (eye, ears, etc.)
- Human augmentation (exoskeleton)

This field has an important connection with neuroscience, to develop neural interfaces and sensory-motor coordination systems for the integration of these bionics devices into the human body/brain.

64.15.10 Health Care and Quality of Life

Health care and quality-of-life robotics is certainly a very promising field, where progress will be directly measured by the well being of people. It is also the best way to promote robotics among the public, especially amongst aging populations.

Surgical robotics allows minimally invasive surgery, which can reduce patient recovery time, and may also improve accuracy and precision. Robotics systems increase the precision of microsurgery and enhance the performance of complex therapies. Surgical robots can restore a surgeon's dexterity. Robotic surgery is also applied to very delicate neurological procedures that are practically impossible to perform without robotic assistance.

Assistive technology will help many people to conduct a more independent life.

Biorobotics, while enhancing the quality of life after diseases or accidents, provides tools for studying biological behavior and brain functions, and is a test bed for the study and evaluation of biological algorithms and modeling.

From the social and ethical standpoint, this is one of the fields in robotics that suffers from the most difficult safety and ethical problems. From a technical point of view, scientists in robotic surgery are working on the problems of reduced dexterity, workspace, and sensory input and possible fatal trouble, which could originate from the breakdown of surgical robot systems. Issues of size, cost, and functionality should also be addressed in surgery, haptic, and assistive robotics.

In the context of assistive technology, some questions concerning the relationship between patients and the health structures in which they are treated can be posed. Are we going to mechanize hospitals and to dehumanize our patients? Shall we improve our health structures, where human nurses can care for patients? May we not develop new psychological and physical dependences?

As a general principle of awareness, we should underline that the high cost of robotic systems in the medical field could widen the digital divide between developed and developing countries, and between layers of the same population.

The field of implantations raises concerns related to the fact that direct brain interfaces may at the same time pose ethical questions related to the enhancement of human function.

The BioX program at the University of Stanford, and the Stanford Center for Biomedical Ethics, funded

a pilot study in this domain called cross-cultural considerations in establishing roboethics for neuro-robot applications [64.62, 63]. This study explores funding mechanisms to investigate the span of ethical issues currently confronting direct brain interface investigators, how different kinds of interfaces may indicate different approaches to bioethics, and how other stakeholders in the deployment and use of this technology (for example, from law, government, and healthcare provider professions) perceive the relative importance of the various bioethics issues for the variety of interfaces that currently exist and those on the horizon.

64.15.11 Military Robotics

Intelligent Weapons

This field includes all devices resulting from the development of traditional military systems using robotics technology (automation, artificial intelligence, etc.):

- Integrated defense systems: an **AI** system for intelligence and surveillance, controlling weapons and aircraft capabilities
- Autonomous tanks: armored vehicles carrying weapons and/or tactical payloads
- Intelligent bombs and missiles
- Unmanned aerial vehicles (**UAVs**): unmanned spy planes and remotely piloted bombers
- Autonomous underwater vehicles (**AUVs**): intelligent torpedoes and autonomous submarines

Robot Soldiers

Humanoids will be employed to substitute humans in performing *sensitive* tasks and missions in environments populated by humans. The main reasons for using humanoids are to permit a one-by-one substitution, without modifying the environment, the human-human interaction, or the rules of engagement. This could be required where safeguarding human life is considered a priority in many different scenarios:

- Urban terrain combat
- Indoor security operations
- Patrolling
- Surveillance

Outdoor security robots could be able to make their night watch rounds and even chase criminals, directed by a remote-control system via an Internet connection or moving autonomously via their own artificial intelligence systems.

Superhumans

There are several projects aimed at developing a superhuman soldier. Actually, the human body cannot perform a task with the same strength, speed, and fatigue resistance as machines. Robotic *augmentation* describes the possibility of extending existing human capabilities through wearable robot exoskeletons, to create superhuman strength, speed, and endurance, including applications such as:

- artificial sensor systems
- augmented reality
- exoskeletons

The benefits of military robots are:

1. tactical/operational strength superiority
2. unemotional behavior, potentially more ethical than humans
3. limiting the loss of human lives in the robotized army
4. better performance of superhuman over human soldiers

Problems could arise from:

1. the inadequacy to manage the unstructured complexity of a hostile scenario
2. the unpredictability of machine behavior
3. the assignment of liability for misbehavior or crimes
4. the increased risk of starting a videogame-like war, due to the decreased perception of its deadly effects

From the human point of view, humans in mixed teams could face psychological problems, such as the practical and psychological problems of having to distinguish between humans from robots and the stress and dehumanization of superhuman soldiers.

In 2007, the Georgia Tech Mobile Robot Lab – lead by Ronald Arkin – led an online *opinion survey on the use of robots capable of lethal force in warfare*. the opinion survey is part of an important research project under a grant from the Army Research Office. The goal of this survey was to determine how acceptable the robots capable of lethal force in warfare are to different people of varying backgrounds and positions.

Military robotics should be thoroughly examined by specialized international organizations, as happens for every type of military technology, to be regulated by international conventions or agreements [64.64].

64.15.12 Educational Robot Kits

The beneficial applications of robotics in education are known and documented.

Robotics is a very good tool for teaching technology (and many other subjects) whilst, at the same time, always remaining very tightly anchored to reality. Robots are real three-dimensional objects which move in space and time, and can emulate human/animal behavior; but, unlike video games, they are real machines, true objects, and students learn much more quickly and easily if they can interact with concrete objects as opposed to formulas and abstract ideas.

In the age of electronics, computers, and networks, it is necessary to modernize not only educational content and tools, but also the methods used in traditional schools.

It is also important to consider that the lifestyle of young people has changed as well as the communication tools they use in their free time. Today, young people communicate via the Internet and mobile telephones using e-mail, SMS, and chat rooms, which allow them to be continually connected to a global community that has no limits regarding location and time.

Young people spend more time playing videogames, playing with their mobile phones or downloading files from the Internet. These activities provide them with experiences that are by now at the same standard as the most sophisticated technological systems. All this has accelerated the pace of life; so much so that fruition and consumption of experiences are both real and virtual. In fact, we are entering the age of cyberspace, which will not replace normal life relationships, but will certainly alter their characteristics.

In this context, we need to consider that traditional teaching and classical tools of support (books, documentaries) are at risk of becoming unsuitable when compared with the everyday possibilities offered to these young people by the world of mass media. Therefore, it is necessary to begin to plan new ways to transmit knowledge which exploit the potential of this new technology.

Learning about robotics is important not only for those students who want to become robotics engineers and scientists, but for every student, because it provides a strong method of reasoning and a powerful tool for grappling with the world. Robotics collects all the competencies needed for designing and constructing machines (mechanics, electrotechnics, electronics), computers, software, communications systems, and networks. The special features of robotics boost student creativity, communication skills, cooperation, and teamwork.

Learning about robotics promotes students' interest in and commitment to traditional basic disciplines (mathematics, physics, technical drawing). Robotic

construction kits, which can combine the physical building of artifacts with their programming, can foster the development of new ways of thinking that encourage new reflections on the relationship between: (1) life and technology, (2) science and its experimental toolset, and (3) robot design, and values and identity.

64.15.13 Robot Toys

The Aibo robot is the Sony's robotic puppy dog with a software-controlled personality and abilities. The entertaining robot, which costs upwards of \$2000, can dance, whimper, guard, and play, developing personalities based on interaction with its owners. Sony has sold over 150 000 Aibos since launching the product in May 1999.

Company officials said that there was a real effort this time to make the Aibo's movements more doglike; designers even studied the way dogs move. Developers replaced a relatively un-dog-like sideways head motion of one motor (as with the previous model, there are 20 motors) with a sort of forward-and-down movement.

Robot toys can be intelligent toys: they can be specifically designed to stimulate children's creativity and the development of their intellectual faculties. They can become children's companions, and – for only children – could play the role of *friends*, *brothers*, or the traditional *imaginary fiend*. They could also be used in the pedagogical assistance of autistic children.

On the negative side of technology, robot toys could cause psychological problems, such as:

- lost touch with the real world
- confusion between the natural and the artificial
- confusion between the real and the imaginary
- technology addiction [64.65]

64.15.14 Entertainment Robotics

Robots will enable the construction of real environments that could either be the perfect (or scaled) copies of some existing environments, or the reconstruction of settings existed centuries/millennia ago, and which we can populate with real or imaginary animals.

Robots and robotics settings will make it possible to build natural phenomena and biological processes, even cruel ones, without involving living beings.

In these settings, the users/audience could live interactive experiences, which are *real*, not only *virtual*.

As extraordinary theatrical machines, robots will develop ever more *real* special effects.

Entertaining robots are already used to display and advertise corporate logos, products, and events. These are marketing tools showed off by the manufacturers on special occasions.

In this framework, we should also consider sexual robots, which will be an important market. They could be used as sexual partners in many fields, from therapy to prostitution, and their use could decrease sexual exploitation of women and children [64.66]. This also raises issues related to intimacy/attachments, and about safety and reliability.

64.15.15 Robotic Art

The role of robotics in contemporary art, along with all the types of interactive artistic expressions (telecommunications, and interactive installations), is gaining importance and success.

Artists are employing advanced technologies to create environments and works of art, utilizing the actuators and sensor to allow their robots to react and change in relation to viewers.

Robotic art will spread because:

- It recalls (and it is inspired by) the mythological traditions of various cultures. These traditions have created fantastic synthetic creatures;
- Robots exert on the population at large a special fascination;
- Robots can be used as tools in artwork and enable the building of artistic expression in shorter times, thus expanding the borders of human creativity;
- Robots can also perform actor's rules and allow playing living art.

The social and individual problems that can be produced by robotic art are, on the one hand, the dissemination of misinformation (by spreading of false information using technology), while on the other hand, technology may prevail over creativity.

64.16 Conclusions and Further Reading

In this chapter we have analyzed the main social and ethical issues in robotics, five years after the birth of roboethics, and after three years of wide and intense international discussion. In the conclusions, we develop some assessments, foresee lines of progress, and give some indications for those who wish to study the subject of roboethics in more depth.

The so-called robotics invasion has not yet been unleashed. Surely, the recent figures of the World Robotics Report (Unesco/Fir 2005) show a steady growing trend of the robotics production and sales. However, often the media demand more inventions and *gadgets* from the robotics laboratories than the laboratories can afford and, looking at the many automatons that are still struggling to walk, the latter's efforts have so far proved to be something of a disappointment. This is certainly a problem and a pressure for the robotics scientists. For the time being, robotics is a field of research and development that can be applied in, and depends, a high level of technology.

However, we are witnessing a true, growing interest in robots from the general public, who are often more excited than the insiders, whose feelings swing between a position of cultural indifference to a behavior dictated by external pressures, be they political or industrial. We are also noticing the modern change – which had already happened in the 1970s in the field of computer science – of the transformation of the robot from a research platform and a working tool to a consumer item, and an object of entertainment. This is a juvenile phenomenon, as shown by the increase of robotics contests among high-school students. Today's young people who are getting their hands on robotics kits will be the robotics professionals and consumers of tomorrow.

Growing interest in the social effects of robotics is easy to observe among international professional associations and orders, stretching over the sister fields of computer ethics and bioethics.

Certainly, roboethics is still far from being a well-established applied ethics, and by *well established* authors mean that it should demonstrate two qualities: to be universally accepted and standardized, or at least adopted by some communities, relevant in size and in political/economic/cultural influence, and to be embodied in the design, production, and use of robots.

In this chapter, we have mentioned two important steps in this general direction: the guidelines for the use of robots in the human environment, drawn up by the ad hoc group of the Japanese METI; and the Roboethics

Charter, which is still in progress, being edited by the appointed committee of the Republic of South Korea. We should recall a few other projects that are studying the effects of the application of robotics to the neurosciences [64.58] and to bioethics/biorobotics [64.67,68]. However, there is no question that we are still at an initial stage of the subject's development.

In fact, considering the history of the two widely applied and structured ethics which are extensively studied and which reach a certain organic unity, bioethics and computer and information ethics [64.46], we acknowledge that their development, which has been happening for over 30 years, came about through leaps and contradictions, chasms and bends, and that they are far from being a suitable ethical standard shared by a plurality of subjects. Both these ethics were born in a policy and legislative vacuum, as technological changes outpaced ethical developments, bringing about unanticipated problems [64.47].

The standardization of roboethics requires the accomplishment of some fundamental steps, both culturally and institutionally. From the general standpoint, it demands that the application of robotics to the human environment, especially to sensitive areas of the human life, will be accepted by the quasi totality of cultures, as has happened with other techno-scientific innovations such as electricity and computer systems. (In the case of free access to the Internet, the issue is still questionable in many nations.) Should this be achieved, roboethics would have already passed the phase of being adjusted to fit different answers and situations, to being modified to the point of having acquired the capability of adapting to different points of views. Different cultures and religions regard the intervention in sensitive fields such as human reproduction, neural therapies, implantations, and privacy differently. These differences originate from cultural specificities regarding fundamental issues, for example, the limit between a human and a cyborg; the separation between the natural and the artificial; the difference between human and artificial intelligence; the border between privacy and the traceability of actions; the concept of integrity and the unity of the human being; the acceptance of diversity (in gender, ethnicity, minorities, etc.); the boundary between replacement and human enhancement; and so on [64.49]. These are all milestones in defining the underlying paradigms, which in turn influence the day-by-day behavior of everyone.

There are many different aspects to be looked at, for instance, in some cultures the reproduction of the

human figure is forbidden. In others, the difference between human and nonhuman is not so sharp. The application of humanoid robots should be set against this background [64.69]. The diversity of ideas on these issues, such as natural versus artificial or animate versus inanimate, has immediate effects on the field of organ transplants, and subsequently of robotic organ implants. As a matter of fact, the debate on human enhancement versus rehabilitation is very active in Europe and the United States, for the time being mainly in the field of bioethics.

From the experience provided by more than 30 years of discussions and disputes in the fields of bioethics and information ethics, we know that all the achievements in the field of science and ethics are neither easy nor negligible.

For those who wish to thoroughly investigate some elements of philosophy of science; of history of science and ethics; of science and engineering's ethics; of the law applied to science and technology, we now suggest some fundamental steps.

In the field of the moral theories related to science and technology, we mention the considerable work of Tom L. Beauchamp from the Kennedy Institute of Ethics [64.49].

Two important annual gatherings of Computer Philosophy, CEPE, Computer Ethics Philosophical Enquiry and IACAP International Association for Computing and Philosophy, to mention just two, have recently added *roboethics* as one of their key topics.

We also encourage students and scholars to consult the works and website of the renowned Center for Computing and Social Responsibility (CCSR) of DeMontfort University, Leicester, UK. The CCSR is internationally recognized for its applied research expertise on the risks and opportunities of information technology. It also organizes the International Conference on the Social and Ethical Impacts of Information and Communication Technology (ETHICOMP) every year.

Furthermore, it is very useful to follow the activity of the regulatory bodies entitled to deal with the issues of science and ethics. In accordance with what was said in Sects. 64.4, 64.5, 64.12, and 64.13 of this chapter, the person's interest should start from the general principles that are essentially accepted by most of the worlds' Nations (at least, nominally), and to come down to the specific applications in our field.

The Ethics of Science and the Technology Programme is part of UNESCO's Division of the Ethics of Science and Technology in the Social and Human Sciences Sector. COMEST is an advisory body to UN-

ESCO. The two bodies work to apply in science and technology the principles of the Universal Declaration of Human Rights.

The Unesco's Ethics of Science and Technology Programme was created in 1998 along with COMEST to provide ethical reflection on science, technology, and their applications. Currently, in accordance with Decision 3.6.1 of the 169th session of the Executive Board, UNESCO is initiating standard-setting action by drafting studies on some new technological areas.

Another body whose activity is useful to follow is the European Group on Ethics in Science and New Technologies and the Forum (EGE, European Group on Ethics in Science and New Technologies), established by the European Commission. The EGE is an independent, pluralist and multidisciplinary body that advise the European Commission on ethical aspects of science and new technologies, regarding the preparation and implementation of community legislation or policies. The forum has many complementary roles. The former body is appointed to provide high-level specialist ethical advice to the European Commission, particularly in relation to the policy arena. The latter was set up under the Framework Programme as a networking activity with the aim of sharing information and exchanging best practices on issues of ethics and science. They work on the basis of the Lisbon Declaration 2000 and the charter of fundamental rights of the European Union approved by all the member states in 2001 (Nice, France).

Concerning the role of science and technology in law, politics, and the public policy in modern democracies, there are important differences between each of the European, the American, and the – we could say – oriental approach. In the United States, the general attitude is definitely more science-based than it is in Europe. In the former case, science is said to speak the truth, and the regulatory process is based more on objective scientific data than on ethical considerations. At the same time, the subjective point of view is taken up by the courts, which are now also intervening directly in areas such as risks to society and scientific knowledge, although the current conceptual tools of jurisprudence in the field of science and technology are still very limited. Nonetheless, in the Anglo Saxon culture, *law does not speak the language of science* [64.70].

On the other hand, in Europe, against the backdrop of the ongoing process of European cohesion, regulation and legislation of science and technology is assuming the character of the foundation of a new political community – the European Union – which is centered around the relationship between science and its applications, and the

community formed by the scientists, technology producers, and citizens. We can safely assume that, given the common classical origin of jurisprudence, the latter process could be helpful in influencing other cultures, for instance, the moderate Arab world.

On the subject of science, technology, and law in America and Europe, we recommend the impressive work by Sheila Jasanoff (Kennedy School of Government Faculty, Harvard University) whose research pivots on the role of science and technology in the law, politics, and public policy in Europe, the United States, and India, with particular reference to the behavior of the American courts in the regulation of science, and to the role of experts.

There is a third way to approach issues in science and society, which could be called oriental. In fact, in Japan and in the Republic of South Korea, issues of robotics and society have been handled more smoothly and pragmatically than in Europe and in America. Due to the general confidence from these societies towards the products of science and technology, the robotics community and the ad hoc ethical committees inside these governments have started to draw up guidelines for the regulation of the use of robotic artefacts. This nonideological nonphilosophical approach has its pros and cons, but it could encourage scientists and experts in Europe and the United States to adopt a more normative position.

For those who are interested in keeping up to date on these issues, a good habit to acquire is to consult the archives and websites of the academic institutions, private associations, and professional orders where problems of science and ethics are followed on a regular basis. The Nobel Prize Pugwash Conference for World Affairs is the umbrella association for this community and NGOs concerned with these issues. The [IEEE Robotics & Automation Society's Technical Committee on Roboethics](#) was formed for the purpose of promoting and collecting research on robotics and society.

The issue of the influence and the pressure provided by the market on science and R&D is handled by applied ethics, and is known as business ethics. In this framework, corporate social responsibility is one of the ways in which enterprises can affirm ethical principles and values. This view was introduced to the United States 15 years ago (especially in the field of health care) and is still running today. Also, training in the responsible conduct of research (RCR) has been adopted by the United States, and is still being applied. The domain of [RCR training](#) does not only include the ethical dimensions of research with human

subjects, but every intricate dimension of responsible conduct in the planning, performance, analysis, and reporting of research. Difficulties here arisen from the small amounts of resources allocated; see also the Organization for Economic Cooperation and Development (OECD) Guidelines for Multinational Enterprises, on the ethical paragraphs.

Concerning the philosophical and epistemological aspect of the issues in ethics and robotics, one of the main problems that people who are interested in roboethics will have to handle is the persistent confusion – ontologically, but especially linguistically – between human and artificial intelligence, as well as between other fundamental concepts of perception, consciousness, self consciousness, emotions, and so on, as applied to humans and to machines.

It must be clarified that the contemporary roboethics is human ethics as applied to robotics, which is considered nonhuman. A strong base of human roboethics is needed in order to responsibly construct the foundations of the final question, which nobody can yet answer: can robots ever become *human*?

This triaging choice, far from rendering the problem simple, renders it technically manageable and fertile of solutions useful to robotics and to society.

At the same time, the need for serious and thorough work into the concepts of intelligence, knowledge, conscience, autonomy, freedom, free will etc. is highlighted. Indeed, the heterogeneous composition of specialists often leads to incessant discussions about the meaning of words, rather than on the content of the myriad, often pressing, issues that need to be faced.

This work is in fact one of the philosophies of robotics, aiming to better define the scientific paradigm. It is important work, as robotics faces the ideal challenge of recreating life artificially and synthetically, which imposes a reopening of discussion and, in some cases, a need to redefine seemingly simple concepts, as well as the need to create new concepts. All this takes place in a multicultural context, which fuels vastly different philosophical backgrounds.

All of this leads to the necessity for the international robotics community to become the author of its own destiny, so as to face directly the task which needs defining, whilst collaborating with academics in the fields of philosophy of law, and generally with experts from the human sciences, to engage with the ethics and social aspects of their research and the applications of the former. Nor should they feel relegated to a mere technoscientific role, delegating to others the task of reflecting and taking action on moral aspects. On the other hand,

a closed-shop attitude would be damaging to the development of robotics, given the interdisciplinary nature of the much of the research undertaken in the field.

From this point of view, roboethics cannot fail to be beneficial to robotics, framing research in close connection with end users and society, and so avoiding many problems that other *sensitive* fields are now facing.

All this, and more, will be wishful thinking if engineering study curricula do not include subjects such as scientific philosophy, history of science, law, and the politics of science, as is already happening in some advanced polytechnics. Once again, we have to say that the deeper study of the history of science, for example, especially in the 19th and 20th centuries, cannot but aid a better understanding of that complex scientific galaxy which is robotics. Even a restricted knowledge of cybernetics and computer science, from Wiener, to von Neumann, to Weizenbaum, will immediately and directly demonstrate that these scientists immediately took care of the ethical and social aspects of their dis-

coveries and realizations, which marked the beginning of the field of computers and robotics.

At the same time, it is necessary that those not involved in robotics keep up to date with the field's real and scientifically predictable developments, in order to base discussions on data supported by technical and scientific reality, and not on appearances or emotions generated by science fiction. In particular, apart from this handbook, one must look to serious magazines published by recognized scientific associations, and not rely on headlines about ambiguous and scandalizing creations that do not really exist.

Ethics is a 1000-year-old human science with an impressive literature. Its application to the field of science technology is no doubt more recent, even though precedents such as the Hippocratic oath suggest an extremely ancient origin. Research on robotics is throwing light on manifold issues across science and the humanities. No wonder it will also open new and unexpected field of studies and application in ethics.

References

- 64.1 R. Brooks: *Flesh and Machines. How Robots will Change us* (Pantheon, New York 2002)
- 64.2 D.S. Landes: *The Unbound Prometheus: Technological Change and Industrial Development in Western Europe from 1750 to the Present* (Cambridge Univ. Press, Cambridge 2003)
- 64.3 R. Kurzweil: *The Age of Spiritual Machines: When Computers Exceed Human Intelligence* (Viking, New York 1999)
- 64.4 R. Cordeschi: *The Discovery of the Artificial: Behavior, Mind and Machines Before and Beyond Cybernetics* (Kluwer Academic, Dordrecht 2002)
- 64.5 Th. Kuhn: *The Structure of Scientific Revolutions* (University of Chicago Press, Chicago 1962)
- 64.6 R. Capurro: Ethical challenges of the information society in the 21st century, *Int. Inform. Library Rev.* **32**, 257–276 (2000)
- 64.7 K. Capek: *R.U.R. Rossum's Universal Robots* (Dover, Mineola 1921, 2001)
- 64.8 B. Joy: Why The Future Doesn't Need us. *Wired* n. 8 (2000)
- 64.9 M. Atiyah, R. Benjamin, A. M. Cetto, M. Meselson, J. Rotblat: Misuse of Science. Eliminating the Causes of War. Address at the 50th Pugwash Conference On Science and World Affaire (Queens' College, Cambridge 2000)
- 64.10 D. Dennet: When HAL kills, Who's to blame?. In: *HAL's Legacy: Legacy: 2001's Computer as Dream and Reality*, ed. by A.C. Clarke (MIT Press, Cambridge 1997)
- 64.11 P. Menzel, F. D'Aluisio: *Robo sapiens: Evolution of a new species* (MIT Press, Cambridge 2000)
- 64.12 H. Moravec: When will computer hardware match the human brain?, *J. Transhumanism* **1** (1998)
- 64.13 L. Floridi, J.W. Sanders: *On the Morality of Artificial Agents, Information Ethics Groups* (University of Oxford, Oxford 2001), <http://web.comlab.ox.ac.uk/loucl/research/areas/ieg/publications/articles/lomaa.pdf>
- 64.14 D.G. Johnson: *Computer Ethics* (Prentice-Hall, New York 2001)
- 64.15 C. Lang: Ethics for artificial intelligence. Wisconsin State-Wide Technology Symposium, Promise or peril? Reflecting on computer technology: Educational, psychological, and ethical implications (2002)
- 64.16 D.B. Parker, S. Swope, B.N. Baker: *Ethical Conflicts in Information and Computer Science, Technology, and Business* (QED Information Sciences, Wellesley 1990)
- 64.17 G. Veruggio: The EURON Roboethics Roadmap, Humanoids'06, December 6, 2006, Genoa, Italy (2006)
- 64.18 W. Wallach: Robot Morals: Creating an Artificial Moral Agent (AMA). 2002 (1998)
- 64.19 J. Gips: Towards the ethical robot. In: *Android epistemology*, ed. by K. Ford, C. Glymour, P. Hayes (AAAI, Menlo Park 1995) pp. 243–252
- 64.20 C.P. Snow: *The two cultures: and a second look* (Cambridge Univ. Press, Cambridge 1993)

- 64.21 K.R. Popper: *The Logic of Scientific Discovery* (Routledge, Abingdon 1959,2002)
- 64.22 S. Lemm: *Summa technologiae* (Suhrkamp Publishers, Frankfurt a.M. 1964)
- 64.23 B. Sterling: Robots and the rest of us. In: *Wired* 12(5) (2004)
- 64.24 J.M. Galván: On Technoethics, *IEEE-RAS Mag.* 10, 58–63 (2003/4)
- 64.25 P. Danielson: *Artificial Morality: Virtuous Robots for Virtual Games* (Routledge, Abingdon 1992)
- 64.26 Storrs Hall J: *Beyond AI: Creating the Conscience of the Machine* (Prometheus Book, New York 2007)
- 64.27 M. Negrotti: *Naturoids. On the Nature of the Artificial* (World Scientific, New Jersey 2002)
- 64.28 I. Asimov: *Runaround. Astounding Science Fiction* (Republished in *Robot*, Doubleday, Garden City 1991)
- 64.29 I. Asimov: *I Robot* (Doubleday, Garden City 1950)
- 64.30 J. M.Galván: The relationship between human beings and technology: theological issues, Seminar at Scuola Superiore Sant'Anna, Pisa, Italy, March 2001
- 64.31 G. Veruggio: Views and visions in Robotics. Hearing at the Italian Senate's 7th Permanent Commission (Rome 2002)
- 64.32 G. Veruggio: *Marine Robotics and society – a global interdisciplinary approach to scientific, technological and educational aspects, Proceedings of the IARP, IWUR2005* (Pisa Univ. Press, Pisa 2005)
- 64.33 S. Blackburn: *Oxford Dictionary of Philosophy* (Oxford Univ. Press, Oxford 1996)
- 64.34 P. Newall: <http://www.galilean-library.org/int11.html> (2005)
- 64.35 H. LaFollette: *(The) Blackwell Guide to Ethical Theory* (Blackwell, New York 1999)
- 64.36 M.S. Gazzaniga: *The Ethical Brain* (Dana, New York 2005)
- 64.37 S. Jasanoff: *Designs on Nature: Science and Democracy in Europe and the United States* (Princeton Univ. Press, Princeton 2005)
- 64.38 P. Singer: *Applied Ethics* (Oxford Univ. Press, Oxford 1986)
- 64.39 H.T. Engelhardt Jr: *The Foundations of Bioethics* (Oxford Univ. Press, New York 1994)
- 64.40 K. Evers: *Codes of Conduct, Standards for Ethics in Research*, European Commission, Directorate-General for Research, Directorate C (Science and Society, Unit C.3, Ethics and Science)
- 64.41 H.B. Shim: Establishing a Korean Robot Ethics Charter, IEEE-ICRA07 Workshop on Roboethics, Rome, 14 April (2007), <http://www.roboethics.org/icra07/contributions/slides>
- 64.42 F. Conway, J. Siegelman: *Dark Hero of the Information Age: In Search of Norbert Wiener, the Father of Cybernetics* (Basic Books, Jackson 2005)
- 64.43 N. Wiener: *Cybernetics*, 2nd ed.: *or the Control and Communication in the Animal and the Machine* (MIT Press, Cambridge 1948,1965)
- 64.44 N. Wiener: *The Human Use of Human Beings: Cybernetics and Society* (Doubleday Anchor, New York 1954)
- 64.45 J. Weizenbaum: *Computer Power and Human Reason: From Judgment to Calculation* (Freeman, New York 1976)
- 64.46 T.W. Bynum, S. Rogerson: *Computer Ethics and Professional Responsibility* (Blackwell, New York 2004)
- 64.47 J. Moor: What Is Computer Ethics?. In: *Computer and Ethics*, ed. by T.W. Bynum (Blackwell, New York 1985)
- 64.48 Stanford Encyclopedia of Philosophy SEP, ed. by Stanford University's Center for the Study of Language and Information (<http://plato.stanford.edu/>)
- 64.49 T. Beauchamp, J. Childress: *Principles of Biomedical Ethics* (Oxford Univ. Press, Oxford 2001)
- 64.50 V.R. Potter: *Bioethics, A bridge to the future* (Prentice-Hall, Englewood Cliffs 1971)
- 64.51 V.R. Potter: Global bioethics: Building on the Leopold legacy. East Lansing: Michigan State University Press. Pollack, J.(2005). Ethics for the Robot Age. *Wired* 13.01 (1988)
- 64.52 P. Whitehouse: *Am J Bioethics* 3(4), W26–W31
- 64.53 J.M. Galván: Technoethics. In *Proceedings, International Conference on Humanoid Robots, IEEE Robotics and Automation Society, Waseda University, Tokyo 22–24 November 2001*
- 64.54 J. Illes, S. Bird: Neuroethics: A modern context for ethics in neuroscience, *Trends in Neuroscience* 29(9), 511–517 (2006)
- 64.55 The Charter of Fundamental Rights of the European Union, De. 7th, 2000 http://www.europarl.europa.eu/charter/default_en.htm
- 64.56 P. Churchland: *A Neurocomputational Perspective* (MIT Press, Cambridge 1989)
- 64.57 B. Reeves, C. Nass: *The Media Equation: How People Treat Computers, Television, and New Media Like Real People and Places* (Cambridge Univ. Press, Cambridge 1966)
- 64.58 J.J. Wagner, D.M. Cannon, H.F.M. Van der Loos: Cross-cultural considerations in establishing roboethics for neuro-robot applications, *Proceedings IEEE International Conf. on Rehabilitation Robotics (ICORR)*, Chicago, IL, June 28–July 1, 2005
- 64.59 J.J. Wagner, D.M. Cannon, D.M. Van der Loos: Cross-Cultural Considerations in Establishing Roboethics for Neuro-Robot Applications Rehabilitation R&D Center, VA Palo Alto Health Care System, Center for Design Research, Stanford University
- 64.60 C.L. Breazeal: *Designing Sociable Robots* (MIT Press, Cambridge 2004)
- 64.61 P. Dario: Biorobotics, *IEEE Robot. Autom. Mag.* 10(3), 4–5 (2003)
- 64.62 Rehabilitation robotics conference ICORR, Chicago (2005)
- 64.63 H.F. Machiel van der Loos: *Cambridge Quarterly of Healthcare Ethics*, 16:303–307 (Cambridge University Press 2007)

64.64

R. Cordeschi, G. Tamburrini: Intelligent machines and warfare: Historical debates and epistemologically motivated concerns. In: *Computing, Philosophy, and Cognition*, ed. by L. Magnani, R. Dossena (College Publication, London 2005)

64.65

Sh. Turkle: *Life on the Screen: Identity in the Age of the Internet* (Touchstone, New York 1995)

64.66

D. Levy: *Love and Sex with Robots: The Evolution of Human-Robot Relationships* (Harper/HarperCollins Publishers, New York 2007)

64.67

P. Dario, M.C. Carrozza, E. Guglielmelli, C. Laschi, A. Menciassi, S. Micera, F. Vecchi: Robotics as a "Future and Emerging Technology: Biomimetics, Cybernetics and Neuro-robotics in European Projects, IEEE Robotics and Automation Magazine, 2005

64.68

T. Ziemke, N. E. Sharkey: Biorobotics. Special issue of *Connection Science*, **10**(3-4) (2002)

64.69

A. Takanishi: Mottainai Thought and Social Acceptability of Robots in Japan. In *Proceedings of the International Workshop on Roboethics, ICRA'07* (International Conference on Robotics and Automation, Rome, 14th of April, 2007)

64.70

S. Jasanoff: Just evidence: The limits of science in the legal process, *J Law Medicine Ethics* **34**(2) (2006)

Acknowledgements

C.20 Inertial Sensors, GPS, and Odometry

by Gregory Dudek, Michael Jenkin

We would like to thank Sarah Jenkin for her help with the figures.

F.42 Industrial Robotics

by Martin Hägele, Klas Nilsson, J. Norberto Pires

The authors are indebted to James Trevelyan of The University of Western Australia for his valuable assistance.

F.44 Aerial Robotics

by Eric Feron, Eric N. Johnson

The authors would like to thank the reviewers for their thorough and useful comments during the development of this chapter. They would also like to acknowledge the support of the US Defense Advanced Research Projects Agency, the US Air Force Office of Scientific Research, and the US Office of Naval Research for their support.

F.48 Robotics in Hazardous Applications

*by James P. Trevelyan, Sung-Chul Kang,
William R. Hamel*

James Trevelyan acknowledges Surya Singh for detailed suggestions on the original draft, and would also like to thank the many unnamed mine clearance experts who have provided guidance and comments over many years, as well as Prof. S. Hirose, Scanjack, Way Industry, and Total Marine Systems for providing photographs.

William R. Hamel would like to acknowledge the US Department of Energy's Robotics Crosscutting Program and all of his colleagues at the national laboratories and universities for many years of dealing with remote hazardous operations, and all of his collaborators at the Field Robotics Center at Carnegie Mellon University, particularly James Osborn, who were pivotal in developing ideas for future telerobots.

Sungchul Kang acknowledges Changhyun Cho, Woosub Lee, Dongsuk Ryu at KIST, Korea for their providing valuable documents and pictures. He also appreciates Munsang Kim for his leading projects that have produced many of research achievements related to Sect. 48.3 enabling technologies.

F.52 Medical Robotics and Computer-Integrated Surgery

*by Russell H. Taylor, Arianna Menciassi,
Gabor Fichtinger, Paolo Dario*

We would like to thank the many colleagues, including clinicians, fellow engineering researchers, and students with whom we have worked over the years. These experiences have provided the essential background that enabled us to write this chapter. In addition, Drs. Dario and Menciassi extend special thanks to their colleagues at the CRIM Laboratory for help in identifying appropriate citations from the rapidly expanding literature within subfields of medical robotics. Drs. Taylor and Fichtinger gratefully acknowledge the support of numerous US Government agencies and other entities that have provided partial funding for many of the research results reported here, with special thanks to the National Science Foundation for generous support of the Engineering Research Center for Computer-Integrated Surgical Systems and Technology under cooperative agreement EEC9731478. Finally, we would like to thank the reviewers for their many constructive comments and suggestions, which have greatly improved this chapter.

G.61 Evolutionary Robotics

by Dario Floreano, Phil Husbands, Stefano Nolfi

We thank the anonymous reviewers and Michael Arbib for constructive suggestions and Mototaka Suzuki for further corrections and editorial help. Dario Floreano was supported by the Swiss National Science Foundation.

About the Authors



Jorge Angeles

McGill University
Department of Mechanical Engineering
and Centre for Intelligent Machines
Montreal, Quebec, Canada
angeles@cim.mcgill.ca

Chapter B.10

Jorge Angeles graduated as an Electromechanical Engineer and obtained the MEng degree in Mechanical Engineering, both at Universidad Nacional Autónoma de México (UNAM), then received the PhD degree in Applied Mechanics from Stanford University. Research activities include robot kinematics, dynamics, design, and control as well as design theory and methodology. Angeles is a Fellow of ASME, CSME, IEEE and RSC, The Academies of Arts, Humanities, and Sciences of Canada, and Honorary Member of IFTOMM, the International Federation for the Promotion of Mechanism and Machine Science.

Gianluca Antonelli

Università degli Studi di Cassino
Dipartimento di Automazione, Ingegneria
dell'Informazione e Matematica
Industriale
Cassino, Italy
antonelli@unicas.it



Chapter F.43

Gianluca Antonelli received the Laurea degree in Electronic Engineering and the Research Doctorate in Electronic Engineering and Computer Science from the University of Naples in 1995 and 2000, respectively. He is currently an Associate Professor at the University of Cassino. His research interests include simulation and control of underwater robotic systems, force/motion control of robot manipulators, multirobot systems, and identification. He is author of the book *Underwater Robots*. He has been a Senior Member of the IEEE since June 2006.

Fumihito Arai

Tohoku University
Department of Bioengineering and
Robotics
Sendai, Japan
arai@imech.mech.tohoku.ac.jp



Chapter B.18

Fumihito Arai received the MS degree from Tokyo University of Science in 1988. He joined Nagoya University in 1989 as a Research Associate. He received the Dr. Eng. from Nagoya University in 1993 and was Associate Professor there from 1998 onwards. Since 2005, he has been Professor at the Department of Bioengineering and Robotics, Tohoku University. His research interests are mainly in micro- and nanorobotics.



Michael A. Arbib

University of Southern California
Computer, Neuroscience and USC Brain
Project
Los Angeles, CA, USA
arbib@usc.edu

Chapter G.62

Michael Arbib is University Professor, Fletcher Jones Professor of Computer Science, a Professor of Biological Sciences, Biomedical Engineering, Electrical Engineering, Neuroscience and Psychology, and Director of the USC Brain Project, all at the University of Southern California.



Antonio Bicchi

Università degli Studi di Pisa
Centro Interdipartimentale di Ricerca
"Enrico Piaggio" e Dipartimento di
Sistemi Elettrici e Automazione
Pisa, Italy
bicchi@ing.unipi.it

Chapter G.57

Antonio Bicchi (Laurea Mechanical Engineering, Pisa 1984, PhD 1989 Bologna) is Professor of Automatic Control and Robotics in the Department of Electrical Systems and Automation of the University of Pisa, Italy. His research interests are in robotics, control, and haptics. He currently serves as the Director of the Interdepartmental Research Center E. Piaggio of the University of Pisa, where he has lead the Automation and Robotics group since 1990. He is an IEEE Fellow and served as Vice President for Member Activities of the IEEE Robotics and Automation Society (2006–2007).

Aude Billard

Ecole Polytechnique Federale de Lausanne (EPFL)
Learning Algorithms and Systems Laboratory (LASA)
Lausanne, Switzerland
aude.billard@epfl.ch



Chapter G.59

Aude Billard is Associate Professor and Head of the Learning Algorithms and Systems Laboratory (LASA) at the School of Engineering at the Ecole Polytechnique Fédérale de Lausanne (EPFL), Switzerland, which she joined in 2002. She is an adjunct faculty member in the Department of Computer Science at the University of Southern California (USC). She received the BS (1994) and MS (1995) degrees in Physics from EPFL. She received her MS in knowledge based systems in 1996 and her PhD in artificial intelligence in 1998 from the University of Edinburgh. Prof. Billard’s research interests include humanoid robotics, human–robot interactions, machine learning, and computational neuroscience.

John Billingsley

University of Southern Queensland
Faculty of Engineering and Surveying
Toowoomba, QLD, Australia
billings@usq.edu.au



Chapter F.46

Over 40 years, John Billingsley has led robotics and mechatronics research in the Universities of Cambridge, Portsmouth, and Southern Queensland, Australia. He holds a Chair of Mechatronic Engineering, is Technology Research Director in the National Centre for Engineering in Agriculture, and has written several textbooks. He was recently awarded an Achievement Medal by The Institution of Engineering and Technology (IET) UK.

Wayne Book

Georgia Institute of Technology
G.W. Woodruff School of Mechanical Engineering
Atlanta, GA, USA
wayne.book@me.gatech.edu



Chapter B.13

Wayne J. Book, HUSCO/Ramirez Distinguished Professor of Fluid Power and Motion Control, has taught at Georgia Institute of Technology since 1974. A Fellow of ASME and IEEE, he holds degrees from M.I.T. (MS, PhD) and The University of Texas (BS) in Mechanical Engineering. His research focuses on system dynamics of fluid power and flexible systems, robotics, and haptic human interfaces.

Cynthia Breazeal

Massachusetts Institute of Technology
The Media Lab
Cambridge, MA, USA
cynthiab@media.mit.edu



Chapter G.58

Cynthia Breazeal is an Associate Professor at the Massachusetts Institute of Technology where she founded and directs the Personal Robots Group in the Media Lab. She is a pioneer of social robotics and human–robot interaction. Her research focuses on developing the principles, techniques, and technologies for socially intelligent robots that interact and communicate with people in humancentric terms, work with humans as peers, and learn from people as an apprentice. She has developed numerous robotic creatures ranging from small hexapod robots, to creating highly expressive humanoid robots. Her recent work investigates the impact of social robots on helping people of all ages.

Oliver Brock

University of Massachusetts
Robotics and Biology Laboratory
Amherst, MA, USA
oli@cs.umass.edu



Chapter D.26

Oliver Brock is an Assistant Professor of Computer Science at the University of Massachusetts, Amherst. He received his Computer Science Diploma in 1993 from the Technical University of Berlin, Germany, and the Masters and PhD degrees in Computer Science from Stanford University in 1994 and 2000, respectively. At the University of Massachusetts Amherst, he is affiliated with the Robotics and Biology Laboratory and the Computational Biology Laboratory. His research focuses on autonomous mobile manipulation and structural computational biology.

Alberto Broggi

Università degli Studi di Parma
Dipartimento di Ingegneria
dell'Informazione
Parma, Italy
broggi@ce.unipr.it



Chapter F.51

Alberto Broggi received the Master degree in Electronic Engineering and the PhD degree in Information Technology both from the University of Parma, Italy. He is now Full Professor at the University of Parma. He is the director of VisLab (the Artificial Vision and Intelligent Systems Lab), whose main milestones are the ARGO Project (a 2000+ km test done on Italian highways in 1998 in which the ARGO vehicle drove itself in automatic mode) and the setup of the Terramax vehicle that reached the finish line of the DARPA Grand Challenge 2005.

Heinrich H. Bülthoff

Max-Planck-Institut für biologische
Kybernetik
Kognitive Humanpsychophysik
Tübingen, Germany
heinrich.buelthoff@tuebingen.mpg.de

Chapter G.63

Professor Heinrich H. Bülthoff is scientific member of the Max Planck Society and Director at the Max Planck Institute for Biological Cybernetics. In his department, an interdisciplinary group of 60 scientists work on psychophysical and computational aspects of higher-level visual processes in object and face recognition, sensory-motor integration, computer graphics psychophysics, and perception and behavior in virtual environments.

Joel W. Burdick

California Institute of Technology
Mechanical Engineering Department
Pasadena, CA, USA
jwb@robotics.caltech.edu

Chapter D.27

Joel Burdick received the PhD degree in Mechanical Engineering from Stanford University. He has been with the department of Mechanical Engineering at the California Institute of Technology since May 1988, where he has been the recipient of the NSF Presidential Young Investigator Award, the Office of Naval Research Young Investigator Award, and the Feynman Fellowship. Professor Burdick's current research interests are in sensor-based robot motion planning, multifingered robotic hand manipulation, and the application of robotics to neural prosthetics.

Wolfram Burgard

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Freiburg, Germany
burgard@informatik.uni-freiburg.de



Chapter E.36

Wolfram Burgard is a Professor for Computer Science at the University of Freiburg and head of the Research Laboratory for Autonomous Intelligent Systems. His research focuses on the development of probabilistic approaches to mobile robot navigation. Over the past years he and his group have developed a series of innovative probabilistic techniques for various mobile robot navigation problems including localization, map building, path planning, and exploration.

Zack Butler

Rochester Institute of Technology
Department of Computer Science
Rochester, NY, USA
zjb@cs.rit.edu



Chapter E.39

Zack Butler received his PhD in Robotics from Carnegie Mellon University in 2000 and is currently an Assistant Professor of Computer Science at the Rochester Institute of Technology. His research interests are in cooperative systems, self-reconfiguring robotics, and the use of robotics in education.

Fabrizio Caccavale

Università degli Studi della Basilicata
Dipartimento di Ingegneria e Fisica
dell'Ambiente
Potenza, Italy
fabrizio.caccavale@unibas.it

Chapter D.29

Fabrizio Caccavale received the Laurea degree and the Research Doctorate degree in Electronic Engineering from the University of Naples Federico II in 1993 and 1997, respectively. He is currently Associate Professor in the Department of Environmental Engineering and Physics of the University of Basilicata. He has published more than 80 journal and conference papers and is co-author of two books. He has been in the program committee of several international conferences and workshops.



Sylvain Calinon

Chapter G.59

Ecole Polytechnique Federale de Lausanne (EPFL)
Learning Algorithms and Systems Laboratory (LASA)
Lausanne, Switzerland
sylvain.calinon@epfl.ch

Sylvain Calinon is a Postdoctoral Fellow at the Learning Algorithms and Systems Laboratory (LASA), Ecole Polytechnique Federale de Lausanne (EPFL), Switzerland. He received his PhD in Robot Programming by Demonstration from LASA, EPFL in May 2007, and his BS/MS in Microengineering with specialization in robotics from the School of Engineering at EPFL in 2003. His research interests cover learning by imitation, human–robot interaction, artificial intelligence, and machine learning.

Guy Campion

Chapter B.17

Université Catholique de Louvain
Centre d'Ingénierie des Systèmes d'Automatique et de Mécanique Appliquée
Louvain-la-Neuve, Belgium
guy.campion@uclouvain.be



Guy Campion received the Masters and the PhD degree in Mechanical Engineering from Université Catholique de Louvain, Louvain-la-Neuve, Belgium. Since 1977 he is Research Associate (Fonds National de la Recherche Scientifique), and part-time Professor with the Centre for Systems Engineering and Applied Mechanics (CESAME) at the same University. His main research interests are in nonlinear control theory, adaptive control, modeling, and control of mechanical systems, especially in nonholonomic systems such as mobile robots.

Raja Chatila

Chapter A.9

For biographical profile, please see the section “About the Part Editors” on page XXII.

François Chaumette

Chapter C.24

INRIA/IRISA
Campus de Beaulieu
Rennes, France
francois.chaumette@irisa.fr



François Chaumette graduated from Ecole Nationale Supérieure de Mécanique, Nantes, France, in 1987. He received the PhD degree in Computer Science from the University of Rennes, France, in 1990. Since 1990, he has been with INRIA at IRISA in Rennes, where he is now Directeur de Recherches and Head of the Lagadic Group. His research interests include robotics and computer vision, especially visual servoing and active perception.



Stefano Chiaverini

Chapter B.11

Università degli Studi di Cassino
Dipartimento di Automazione, Ingegneria dell'Informazione e Matematica Industriale
Cassino, Italy
chiaverini@unicas.it

Stefano Chiaverini is a Professor of Automatic Control. His research interests include inverse kinematics techniques, redundant manipulator control, cooperative robots, force/position control, underwater robotics, and mobile robotic systems. He has published about 150 papers, is co-editor of the book *Complex Robotic Systems* and is co-author of the book *Fondamenti di Sistemi Dinamici*. He is a Senior Member of the IEEE.



Nak Young Chong

Chapter D.32

Japan Advanced Institute of Science and Technology (JAIST)
School of Information Science
Ishikawa, Japan
nakyong@jaist.ac.jp

Nak-Young Chong is an Associate Professor at the Japan Advanced Institute of Science and Technology. He received the PhD degree from Hanyang University, Seoul, Korea, in 1994. He was on the research staff at Daewoo Heavy Industries, Korea, and AIST, Japan, from 1994 to 2003. He has been chair of Fujitsu Robotics Work Group and Robot Information Processing Work Group since 2004. His research interests cover human-friendly robots, teleoperation, intelligent mechanical systems, and knowledge networking robot control.

Howie Choset

Chapter F.50

Carnegie Mellon University
The Robotics Institute
Pittsburgh, PA, USA
choset@cs.cmu.edu



Howie Choset is an Associate Professor of Robotics at Carnegie Mellon University, where he conducts research in motion planning and design of serpentine mechanisms, and coverage path planning for demining. Choset directs the Undergraduate Robotics Minor at Carnegie Mellon and teaches an overview course on Robotics which uses series of custom-developed Lego Labs to complement the coursework.

Henrik I. Christensen Chapter A.4

For biographical profile, please see the section “About the Part Editors” on page XXI.

Wankyun Chung

POSTECH
Department of Mechanical Engineering
Pohang, Korea
wkchung@postech.ac.kr


Chapter A.6

Wankyun Chung received his BS degree from Seoul National University in 1981, his MS degree in 1983, and his PhD from KAIST in 1987. He is a Professor in the School of Mechanical Engineering, POSTECH and Director of the National Research Laboratory for Intelligent Mobile Robot Navigation. He has written two books on the control of mechanical systems and published more than 60 journal papers on control and robotics. His research interests include SLAM, underwater robots, and biorobots.



Woojin Chung

Korea University
Department of Mechanical Engineering
Seoul, Korea
smartrobot@korea.ac.kr

Chapter B.17

Woojin Chung received the BS degree from the Seoul National University and the MS and PhD degrees from the University of Tokyo. He is currently an Assistant Professor at the Department of Mechanical Engineering, Korea University. He received the best paper awards from RSJ and IEEE Robotics and Automation Society. He works on design and control of nonholonomic systems and mobile robot navigation.



J. Edward Colgate

Northwestern University
Department of Mechanical Engineering
Segal Design Institute
Evanston, IL, USA
colgate@northwestern.edu

Chapter G.57

Ed Colgate is the Pentair–Nugent Professor in the Department of Mechanical Engineering at Northwestern University, Evanston. He has worked extensively in the areas of haptic interface and teleoperation, and he is the co-inventor of *cobots*, a class of collaborative robots. Dr. Colgate is the founder of two start-up companies and he is the founding Editor-in-Chief of the IEEE Transactions on Haptics.

Peter Corke

Commonwealth Scientific and Industrial
Research Organisation (CSIRO)
ICT Centre
Kenmore, QLD, Australia
peter.corke@csiro.au


Chapter F.49

Peter Corke is a Chief Principal Research Scientist at CSIRO, where his research focus has been on vision-based robot control and field robotics. He is a member of the editorial board of the International Journal of Robotics Research, a founding editor of the Journal of Field Robotics, and a member of the editorial board of the Springer STAR book series.

Jock Cunningham

Commonwealth Scientific and Industrial
Research Organisation (CSIRO)
Division of Exploration and Mining
Kenmore, QLD, Australia
jock.cunningham@csiro.au


Chapter F.49

Jock Cunningham has 20 years mining R&D experience with Mount Isa Mines: instrumentation, process control, automation for mineral processing/smeltering, robotic nondestructive analysis of rocks, and process streams and teleremote control of mobile equipment. He spent 1 year with Science Applications International in the USA and has worked for CSIRO for 10 years in a range of research/management roles. His current interests include the development of a robotically implemented mining method called ROES.



Mark R. Cutkosky

Stanford University
Mechanical Engineering
Stanford, CA, USA
cutkosky@stanford.edu

Chapter C.19

Mark Cutkosky is a Professor in the Design Group of Mechanical Engineering at Stanford University. He received the PhD degree in 1985 from Carnegie Mellon University. His research includes haptics, tactile sensing, and bio-inspired robots. He has graduated 30 PhD students in these areas and is a former Fulbright Faculty Chair, Charles M. Pigott Chair, and NSF PYI awardee.



Kostas Daniilidis

Chapter C.23

University of Pennsylvania
Department of Computer and Information
Science
GRASP Laboratory
Philadelphia, PA, USA
kostas@cis.upenn.edu

Kostas Daniilidis is Associate Professor of Computer and Information Science at the University of Pennsylvania, where he is a member of the interdisciplinary GRASP laboratory. He obtained the BSE and MSE degrees in Electrical Engineering from the National Technical University of Athens (1986) and his Dr. rer.nat. in Computer Science from the University of Karlsruhe (1992). His research interests are in space and motion perception with machines with applications to navigation, panoramic vision, and immersive environments.

Paolo Dario



Chapter F.52

Scuola Superiore Sant'Anna
ARTS Lab e CRIM Lab
Pisa, Italy
paolo.dario@sssup.it

Paolo Dario received his Dr. Eng. degree in Mechanical Engineering from the University of Pisa, Italy, in 1977. He is currently Professor of Biomedical Robotics at the Scuola Superiore Sant'Anna in Pisa. His main research interest is in biorobotics and he is active in projects on the development of mechatronic and robotic systems for rehabilitation, prosthetics, surgery, and microendoscopy. He is the author of more than 200 journal papers, an IEEE Fellow, and a recipient of the Joseph Engelberger Award.

Alessandro De Luca



Chapter B.13

Università degli Studi di Roma "La
Sapienza"
Dipartimento di Informatica e
Sistemistica "A. Ruberti"
Roma, Italy
deluca@dis.uniroma1.it

Alessandro De Luca has been Professor of Robotics at the Università di Roma "La Sapienza" since 2000. He has published about 140 journal/conference papers, mostly on flexible manipulators, kinematic redundancy, underactuated robots, and wheeled mobile robots. Since 2004, he has been Editor-in-Chief of the IEEE Transactions on Robotics. He was General Chair of the 2007 IEEE ICRA. In 2005, he received the Helmholtz Humboldt Research Award for foreign scientists. He is a member of the Search Committee for Technical Science of the Körber European Science Award and an IEEE Fellow.



Joris De Schutter

Chapter A.7

Katholieke Universiteit Leuven
Department of Mechanical Engineering
Leuven-Heverlee, Belgium
joris.deschutter@mech.kuleuven.be

Joris De Schutter obtained the PhD degree in Mechanical Engineering from K.U. Leuven in 1986. His research interests are in sensor-based robot control and programming including hybrid force/vision control and multisensor fusion, and control and optimization of mechanical and mechatronic systems. He is co-author of two volumes in the STAR book series.



Rüdiger Dillmann

Chapter G.59

Universität Karlsruhe
Institut für Technische Informatik
Karlsruhe, Germany
dillmann@ira.uka.de

Professor Rüdiger Dillmann received his PhD in Electrical Engineering from the University of Karlsruhe in 1980. Since 1987 he has been Professor at the Department of Computer Science and Technical Engineering (CSE) and since 2001 Director of the Industrial Applications of Informatics and Microsystems (IAIM) research group. His research interests are in interactive multimodal humanoid robots, bio-inspired robot systems, autonomous ground-based vehicles, computer-supported surgery, and augmented-reality-based simulation techniques.

Lixin Dong



Chapter B.18

ETH Zentrum
Institute of Robotics and Intelligent
Systems
Zürich, Switzerland
ldong@ethz.ch

Lixin Dong received the PhD degree in Microsystem Engineering from Nagoya University in 2003. He became Research Associate, Lecturer, and Associate Professor at Xi'an University of Technology in 1992, 1995, and 1998, respectively, and Assistant Professor at Nagoya University in 2003. He is Senior Research Scientist at ETH Zurich, where his main research interests include nanorobotic manipulation and related technologies including carbon nanotubes, nanofabrication, nano-mechanochemistry, nanoassembly, NEMS, and nanorobotics.

Gregory Dudek

McGill University
Department of Computer Science
Montreal, QC, Canada
dudek@cim.mcgill.ca



Chapter C.20

Gregory Dudek is a Professor in the McGill University School of Computer Science and a member of the McGill Research Centre for Intelligent Machines (CIM). He was named a William Dawson Scholar. He directs the McGill Mobile Robotics Laboratory, where his research interests include perception for mobile robotics, vision-based robotics, computational vision, and collaborative filtering.

Mark Dunn

University of Southern Queensland
National Centre for Engineering in
Agriculture
Toowoomba, QLD, Australia
mark.dunn@usq.edu.au

Chapter F.46

Mark Dunn has just completed his Engineering PhD entitled 'Applications of vision sensing in agriculture' and is investigating many diverse case studies from counting macadamia nuts with machine vision to using cameras to differentiate between sheep, goats, and feral pigs.

Hugh Durrant-Whyte

University of Sydney
ARC Centre of Excellence for Autonomous
Systems
Australian Centre for Field Robotics (ACFR)
Sydney, NSW, Australia
hugh@acfr.usyd.edu.au

Chapter C.25

Hugh Durrant-Whyte received the MSE and PhD degrees, both in Systems Engineering, from the University of Pennsylvania, USA, in 1985 and 1986, respectively. Since 1995 he has been a Professor at the University of Sydney, where he leads the Australian Centre for Field Robotics (ACFR). His research work focuses on autonomous vehicle navigation and decentralized data fusion methods with applications including automation in cargo handling, mining, defence, and marine systems.

Jan-Olof Eklundh

KTH Royal Institute of Technology
Stockholm, Sweden
joe@nada.kth.se



Chapter C.23

Jan-Olof Eklundh is Professor in Computer Science at KTH, Stockholm, where in 1996 he initiated the Center for Autonomous Systems. His research interests cover a broad range of topics in computational vision and robotics. He chaired ICCV 1990 and ECCV 1994. He is a member of the Royal Swedish Academy of Science, the Royal Danish Academy of Sciences and Letters, and the Royal Swedish Academy of Engineering Science.

Aydan M. Erkmen

Middle East Technical University
Department of Electrical Engineering
Ankara, Turkey
aydan@metu.edu.tr



Chapter F.50

Aydan M. Erkmen received the BS degree from Bogazici University, Istanbul, Turkey in Electrical Engineering (1978), her MS degree in Electrical and Computer Engineering from Drexel University, Philadelphia (1981) and her PhD in Information Technology and Engineering from George Mason University, Fairfax in 1989. She is full professor in the Electrical Engineering Department of the Middle East Technical University, Ankara, Turkey. Her research interests include mechatronic design, search, rescue and surveillance robotics, mobile sensor networks, biomimetic manipulation, biologically inspired control in unstructured environments, and uncertainty modeling.

Bernard Espiau

INRIA Rhône-Alpes
Saint-Ismier, France
bernard.espiau@inria.fr

Chapter B.16

Bernard Espiau received a PhD degree in Automatic Control (Nantes, France, 1975). He conducted research projects in robotics, in the areas of sensor-based control, visual servoing, control architectures, and bipedal locomotion. He launched the BIP (biped) robot project completed in 2000. From 2001 to 2007 he was director of INRIA Rhône-Alpes. He is presently Deputy Scientific Director of INRIA.



Roy Featherstone

Chapter [A.2](#)

The Australian National University
Department of Information Engineering
Canberra, ACT, Australia
roy.featherstone@anu.edu.au

Dr. Featherstone received the PhD degree from Edinburgh University in 1984. He is the inventor of the articulated-body algorithm, and the author of two books on rigid-body dynamics. He has 7 years experience in industry and was awarded an EPSRC Advanced Research Fellowship in 1992, which was held at Oxford University. He currently works on robot dynamics, control, simulation, and actuator technology.

Eric Feron



Chapter [F.44](#)

Georgia Institute of Technology
School of Aerospace Engineering
Atlanta, GA, USA
feron@gatech.edu

Eric Feron has been the Dutton/Ducoffe Professor of Aerospace Software Engineering at the Georgia Institute of Technology since 2005. From 1993 to 2005, he was with the Department of Aeronautics and Astronautics at the Massachusetts Institute of Technology. Eric Feron’s work concentrates on control systems and optimization and their applications to aircraft dynamics, aerial robots and their certification, and transportation systems.

Gabor Fichtinger



Chapter [F.52](#)

Queen’s University
School of Computing
Kingston, ON, Canada
gabor@cs.queensu.ca

Gabor Fichtinger received the BS and MS degrees in Electrical Engineering, and doctorate in Computer Science from the Technical University of Budapest, Hungary, in 1986, 1988, and 1990, respectively. Dr. Fichtinger’s primary research area is computer-assisted surgery, with a strong accent on image-guided needle-based interventions. Further research interest is in systems for robot-assisted minimally invasive surgical procedures executed under image guidance. Primarily application areas are in cancer therapy musculoskeletal procedures.



Paolo Fiorini

Chapter [F.50](#)

Università degli Studi di Verona
Dipartimento di Informatica
Verona, Italy
paolo.fiorini@univr.it

Paolo Fiorini received the Laurea degree from the University of Padova (Italy), the MS degree from the University of California at Irvine, both in Electrical Engineering, and the PhD in Mechanical Engineering from the University of California at Los Angeles. From 1985 to 2000, he was with NASA-JPL, Caltech, and since 2000 he has been an Associate Professor of Control Systems at the University of Verona.



Robert B. Fisher

Chapter [C.22](#)

University of Edinburgh
School of Informatics
Edinburgh, UK
rbr@inf.ed.ac.uk

Robert Fisher received the PhD degree from the University of Edinburgh in 1987 and has been, since then, an academic in Edinburgh University’s School of Informatics. His research covers topics in 3-D computer vision and video analysis. He has led the development of several well-used computer vision web resources, including CVonline. He has published six books and about 200 scientific articles.

Paul Fitzpatrick



Chapter [G.56](#)

Italian Institute of Technology
Robotics, Brain, and Cognitive Sciences
Department
Genova, Italy
paul.fitzpatrick@iit.it

Paul Fitzpatrick received the PhD degree in Computer Science from MIT, on developmental approaches to machine perception. He worked at MIT as a postdoctoral lecturer, then held a postdoctoral position at LIRA-Lab at the University of Genoa, Italy. He is now working at the Italian Institute of Technology. His primary interest is the development of machine perception for object manipulation.

Dario Floreano

Ecole Polytechnique Federale de Lausanne (EPFL)
Laboratory of Intelligent Systems
Lausanne, Switzerland
dario.floreano@epfl.ch



Chapter G.61

Dario Floreano is Associate Professor in the School of Engineering at the Swiss Federal Institute of Technology in Lausanne (EPFL), Switzerland. His research activities span evolutionary robotics, swarm robotics, and biomimetic robotics, and evolvable hardware. He has contributed to the development of several biomimetic robots, such as swarms of wheeled and flying robots. He is cofounder of the International Society for Artificial Life, Inc., and a Member of the Board of Governors of the International Society for Neural Networks. He has been an IEEE Senior Member since 2005.

Thor I. Fossen

Norwegian University of Science and Technology (NTNU)
Department of Engineering Cybernetics
Trondheim, Norway
fossen@iee.org



Chapter F.43

Professor Thor I. Fossen received the MS degree in Naval Architecture and PhD degree in Control Systems from the Norwegian University of Science and Technology. Professor Fossen has been instrumental in the development of several industrial autopilot and dynamic positioning systems. He founded the company Marine Cybernetics, which offers service for hardware-in-the-loop (HIL) testing of marine control systems. His work on weather optimal positioning control for marine vessels received the Automatica Prize Paper Award in 2002. Fossen is currently Head of Automatic Control at the Centre for Ships and Ocean Structures (CESOS), Norwegian Centre of Excellence.

Li-Chen Fu

National Taiwan University
Department of Electrical Engineering
Taipei, Taiwan, R.O.C.
lichen@ntu.edu.tw



Chapter A.6

Li-Chen Fu received the PhD degree (1987) from University of California at Berkeley, and then joined the Department of Electrical Engineering of the National Taiwan University. His research interests include robotics, control, scheduling, and home automation. His numerous awards include the Distinguished Research Awards (1995, 1998, 2000) and Distinguished Research Fellowship (2001–2007) from the National Science Council, R.O.C., and the Industry-Academia Collaboration Award (2004) from the Ministry of Education, R.O.C. He has been an IEEE Fellow since 2004.

Maxime Gautier

Université de Nantes
IRCCyN, ECN
Nantes, France
maxime.gautier@irccyn.ec-nantes.fr



Chapter B.14

Maxime Gautier received the Doctorat d'Etat degree in Robotics and Control Engineering from the University of Nantes, France, in 1990. Since 1991, he has been a Professor of Automatic Control at the University of Nantes. He is a member of the Robotics Team, Institut de Recherche en Communication et Cybernétique de Nantes (IRCCyN, UMR-CNRS n6597). His research topics include modeling, identification, and control of robots.

Martin A. Giese

University of Wales
Department of Psychology
Bangor, UK
martin.giese@uni-tuebingen.de



Chapter G.63

Martin A. Giese (PhD in Electrical Engineering), after a postdoc at CBCL (MIT), headed the HONDA Cambridge Research Laboratory. He founded the Laboratory for Action Representation and Learning at the Hertie Institute for Clinical Brain Research, Tübingen, Germany in 2001, and is now Senior Lecturer at the Department of Psychology, University of Bangor, UK. His research addresses learning in action recognition and control.



Ken Goldberg

Chapter D.32

University of California at Berkeley
Department of Industrial Engineering and
Operations Research
Berkeley, CA, USA
goldberg@ieor.berkeley.edu

Ken Goldberg is Professor of Industrial Engineering and Operations Research at UC Berkeley, with joint appointments in Electrical Engineering and Computer Science, and the School of Information. Goldberg was named IEEE Fellow in 2005 and serves (2006–2009) as Vice President of Technical Activities for the IEEE RAS. He is Founding Chair of the IEEE Transactions on Automation Science and Engineering (T-ASE) Advisory Board.



Clément Gosselin

Chapter B.12

Université Laval
Département de Génie Mécanique
Quebec, QC, Canada
gosselin@gmc.ulaval.ca

Clément Gosselin received the PhD degree and the D.W. Ambridge Award from McGill University, Montreal, Quebec, Canada in 1988. Since 1997 he has been a Full Professor in the Department of Mechanical Engineering at University Laval, Quebec and a Canada Research Chair in Robotics and Mechatronics since January 2001. His research interests are kinematics, dynamics, and control of robotic mechanical systems with a particular emphasis on parallel mechanisms and robot hands. Dr. Gosselin is a Fellow of the ASME.

Agnès Guillot

Université Pierre et Marie Curie – CNRS
Institut des Systèmes Intelligents et de
Robotique
Paris, France
agnes.guillot@lip6.fr



Chapter G.60

Agnès Guillot is Assistant Professor of Psychophysiology at Paris X University. She graduated in Human and Animal Psychology, and holds PhDs in Psychophysiology and Biomathematics. Her main scientific interests are in action selection in animals and robots. She coordinates the biomimetic modeling of the Psikharpax project that aims to synthesize an artificial rat.

Martin Hägele

Fraunhofer IPA
Robot Systems
Stuttgart, Germany
mmh@ipa.fhg.de



Chapter F.42

Martin Hägele studied Mechanical Engineering and Engineering Science at the University of Stuttgart and the George Washington University. Since 1989 he is with Fraunhofer IPA and was promoted to Head of the Department Robot Systems in 1993. He received the Award of the German Association of Machine Tools Manufacturers (VDW) in 1990, the Fraunhofer IPA Innovation Award in 1995, the European Innovation and Emerging Technology Award in 1998, and the Engelberger Robotics Award in 2007.



Gregory D. Hager

Chapter A.4

Johns Hopkins University
Department of Computer Science
Baltimore, MD, USA
hager@cs.jhu.edu

Gregory D. Hager is a Professor of Computer Science at Johns Hopkins University. Prof. Hager received the MS and PhD degrees in computer science from the University of Pennsylvania in 1985 and 1988, respectively. His current research interests include visual tracking, vision-based control, medical robotics, and human–computer interaction. He is a Fellow of the IEEE for his contributions in vision-based robotics.



David Hainsworth

Chapter F.49

Commonwealth Scientific and Industrial
Research Organisation (CSIRO)
Division of Exploration and Mining
Kenmore, QLD, Australia
david.hainsworth@csiro.au

David Hainsworth holds the degrees of BE and PhD in Electrical Engineering from the University of Queensland, Australia. For the last 15 years he has been with the CSIRO Division of Exploration and Mining, where he is currently leader of the Mining Automation Research Group. His main research interests are automation for longwall coal mining and sensor development for underground navigation.

William R. Hamel

University of Tennessee
Mechanical, Aerospace, and Biomedical
Engineering
Knoxville, TN, USA
whamel@utk.edu



Chapter F.48

William R. Hamel is a Professor and the Head of the Mechanical, Aerospace, and Biomedical Engineering Department at the University of Tennessee. He worked for 31 years at the Oak Ridge National Laboratory in robotics and remote systems for nuclear, military, and space applications. He is an IEEE Fellow with ongoing research in telerobotics for hazardous environments.

Blake Hannaford

University of Washington
Department of Electrical Engineering
Seattle, WA, USA
blake@ee.washington.edu



Chapter D.30

Blake Hannaford, PhD, is Professor of Electrical Engineering, Adjunct Professor of Bioengineering, Mechanical Engineering, and Surgery at the University of Washington, Seattle. He was awarded the National Science Foundation's Presidential Young Investigator Award, the Early Career Achievement Award from the IEEE Engineering in Medicine and Biology Society, and was named IEEE Fellow in 2005. His currently active interests include haptic displays on the Internet and surgical robotics.

Kensuke Harada

National Institute of Advanced Industrial
Science and Technology (AIST)
Intelligent Systems Research Institute
Tsukuba, Japan
kensuke.harada@aist.go.jp



Chapter G.56

Dr. Harada's main area of research includes mechanics and control of robotic systems such as humanoid robots and robot fingers. He received the PhD degree in 1997 from Kyoto University. From 1997 to 2002, he worked at Hiroshima University. Since 2002, he has been a researcher at AIST. From 2005 to 2006, he was also a visiting scholar at Stanford University.

Martial Hebert

Carnegie Mellon University
The Robotics Institute
Pittsburgh, PA, USA
hebert@ri.cmu.edu



Chapter E.36

Professor Martial Hebert's interests include computer vision, 3-D model building and recognition, and perception for mobile robots. His group has developed approaches for object recognition in images and video sequences, techniques for recognizing 3-D shapes in 3-D point clouds and for building models of 3-D objects, and for modeling the environment of autonomous robots from sensor data, and their integration into systems.

Thomas C. Henderson

University of Utah
School of Computing
Salt Lake City, UT, USA
tch@cs.utah.edu



Chapter C.25

Professor Thomas Henderson received the BS degree in Mathematics from Louisiana State University in 1973 and the PhD degree in Computer Science from The University of Texas at Austin in 1979. Since 1989 he has been Professor of Computer Science at the University of Utah. His current research interests concentrate on smart sensor networks, the simulation of accidental fires and explosions, on the DARPA Urban Challenge, and on technical drawing analysis.

Joachim Hertzberg

Universität Osnabrück
Institut für Informatik
Osnabrück, Germany
hertzberg@informatik.uni-osnabrueck.de



Chapter A.9

Joachim Hertzberg holds diploma and doctorate degrees in Informatics. He is a Full Professor for Informatics/Knowledge-Based Systems at Osnabrück University; previously, he has worked at Bonn University and Fraunhofer AIS. His areas of research are AI, action planning, robot navigation, mapping, and using semantic information in robot control.



Hirohisa Hirukawa

Chapter G.56

National Institute of Advanced Industrial Science and Technology (AIST)
Intelligent Systems Research Institute
Tsukuba, Japan
hiro.hirukawa@aist.go.jp

Dr. Hirukawa is a Deputy Director of Intelligent Systems Institute of AIST and a Professor at Tsukuba University. He developed a humanoid robot HRP-2 and software platform for humanoid robotics OpenHRP. His research interests include humanoid robotics and motion planning. He is a co-author of a textbook “Humanoid Robot”.



Gerd Hirzinger

Chapter D.31

Deutsches Zentrum für Luft- und Raumfahrt (DLR) Oberpfaffenhofen
Institut für Robotik und Mechatronik
Wessling, Germany
gerd.hirzinger@dlr.de

Gerd Hirzinger received his doctor’s degree from the Technical University of Munich in 1974. Since 1992 he has been Director at the DLR Institute for Robotics and Mechatronics. His institute develops innovative robot systems for space and terrestrial applications, telerobotic concepts, man–machine interfaces, sensors and actuators, sensory feedback, ultra-lightweight robot arms, articulated hands, planetary rovers, and medical technology. He was prime investigator of ROTEX, the first remotely controlled robot in space, which flew onboard shuttle COLUMBIA in April 93. He has received numerous national and international awards.

John Hollerbach

University of Utah
School of Computing
Salt Lake City, UT, USA
jmh@cs.utah.edu



Chapter B.14

John M. Hollerbach is Professor of Computing, Research Professor of Mechanical Engineering, and Director of the Robotics Track at the University of Utah. Previously he held faculty positions at MIT and McGill University. He received his PhD from Massachusetts Institute of Technology in 1978. He is Editor of the International Journal of Robotics Research. His research interests include haptic interfaces and virtual reality.

Robert D. Howe

Harvard University
Division of Engineering and Applied Sciences
Cambridge, MA, USA
howe@seas.harvard.edu



Chapter C.19

Robert D. Howe is Gordon McKay Professor of Engineering in the Harvard School of Engineering and Applied Sciences. Dr. Howe’s research interests focus on manipulation, the sense of touch, and haptic interfaces. Biomedical applications include robotic and image-guided surgery. Dr. Howe earned the BA degree in Physics from Reed College and the PhD degree in Mechanical Engineering from Stanford University.



Phil Husbands

Chapter G.61

University of Sussex
Department of Informatics
Falmer, Brighton, UK
philh@sussex.ac.uk

Phil Husbands is Professor of Computer Science and Artificial Intelligence in the Department of Informatics and Co-director of the Centre for Computational Neuroscience and Robotics at the University of Sussex. His research interests include biologically inspired adaptive robotics, evolutionary systems, computational neuroscience, and creative systems.



Seth Hutchinson

Chapter C.24

University of Illinois
Department of Electrical and Computer Engineering
Urbana, IL, USA
seth@uiuc.edu

After receiving the PhD degree from Purdue University, Seth Hutchinson joined the University of Illinois in Urbana-Champaign, where he is currently a Professor in the Department of Electrical and Computer Engineering. He has published more than 150 papers in the area of sensor-based robotics, and is co-author of the books *Principles of Robot Motion: Theory, Algorithms, and Implementations* and *Robot Modeling and Control*.

Adam Jacoff

National Institute of Standards and Technology
Intelligent Systems Division
Gaithersburg, MD, USA
adam.jacoff@nist.gov



Chapter F.50

Adam Jacoff has directed technology readiness level assessments of autonomous mobility for the US Army, developed standard performance test methods for emergency response robots through ASTM International, and chaired international RoboCupRescue robot competitions since 2001. He received the BS degree in Mechanical Engineering from the University of Maryland and the MS degree in Computer Science from Johns Hopkins University.

Michael Jenkin

York University
Computer Science and Engineering
Toronto, Ontario, Canada
jenkin@cse.yorku.ca



Chapter C.20

Michael Jenkin is a Professor of Computer Science and Engineering at York University, Canada. Working in the areas of virtual reality and visually guided autonomous robots, he is the co-author with G. Dudek of *Computational Principles of Mobile*. He has worked on a number of large robotics projects including the ARK and AQUA Robotic Systems and is also one of the designers of IVY – a six-sided immersive projective environment.

Eric N. Johnson

Georgia Institute of Technology
Daniel Guggenheim School of Aerospace Engineering
Atlanta, GA, USA
eric.johnson@ae.gatech.edu



Chapter F.44

Eric Johnson is the Lockheed Martin Associate Professor of Avionics Integration in the School of Aerospace Engineering at Georgia Tech. His research includes fault-tolerant estimation and flight control as well as digital avionics systems. He was awarded the PhD degree from Georgia Tech, the MS degrees from MIT and The George Washington University, and the BS degree from the University of Washington.

Shuuji Kajita

National Institute of Advanced Industrial Science and Technology (AIST)
Intelligent Systems Research Institute
Tsukuba, Japan
s.kajita@aist.go.jp



Chapter B.16

Shuuji Kajita received the Dr. degree in Control Engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1996. In 1985, he joined the Mechanical Engineering Laboratory, National Institute of Advanced Industrial Science and Technology (AIST). During 1996–1997, he was a Visiting Researcher at the California Institute of Technology. Since 2007, he has been the Leader of the Humanoid Robotics Group of Intelligent Systems Research Institute, AIST, Japan.

Makoto Kaneko

Chapter B.15

For biographical profile, please see the section “About the Part Editors” on page XXII.

Sung-Chul Kang

Korea Institute of Science and Technology
Cognitive Robotics Research Center
Seoul, Korea
kasch@kist.re.kr



Chapter F.48

Sung-Chul Kang is a Principal Research Scientist in the Intelligent Robotics Research Center of the Korea Institute of Science and Technology (KIST). His research interests are in intelligent dependable manipulation, field service robot for rescue, military and hazardous application, haptics, tactile interfaces, and small indoor airship design, control, and application. He has a PhD in Mechanical Design and Production Engineering from the Seoul National University, Seoul, Korea.

Imin Kao

State University of New York at Stony Brook
Department of Mechanical Engineering
Stony Brook, NY, USA
imin.kao@stonybrook.edu



Chapter D.27

Being the Director of the Manufacturing Automation Laboratory (MAL) at SUNY Stony Brook, Professor Kao conducts research in robotics and manufacturing, specialized in the modeling of contact interface. He integrates his study with MEMS transducers to study intelligent contact interface that augments the capability of robots and humans. The soft-contact theory he proposed, which correlates the growth of contact radius with normal force, subsumes the well-known Hertzian model.



Lydia E. Kavraki

Chapter A.5

Rice University
Department of Computer Science, MS 132
Houston, TX, USA
kavraki@rice.edu

Lydia E. Kavraki is the Noah Harding Professor of Computer Science and Professor of Bioengineering at Rice University. She earned the PhD degree in Computer Science from Stanford University. Kavraki's research is in physical algorithms and their applications in robotics and structural bioinformatics. Her recent work has focused on motion planning with realistic physical constraints, robot coordination, and hybrid systems. Her awards include the Grace Murray Hopper Award from the Association for Computing Machinery.



Homayoon Kazerooni

Chapter D.33

University of California at Berkeley
Berkeley Robotics and Human
Engineering Laboratory
Berkeley, CA, USA
kazerooni@berkeley.edu

Dr. Kazerooni holds a Doctorate from MIT and is currently a Professor in the Mechanical Engineering Department at the University of California, Berkeley. He is the Director of the Berkeley Robotics and Human Engineering Laboratory. His research activities have spanned many aspects of human-machine systems, control sciences, robotics, orthotics, prosthetics, mechatronics, and electromechanical design. His latest work on lower extremity exoskeletons resulted in the creation of the very first energetically autonomous orthotic lower extremity exoskeleton.

Charles C. Kemp

Chapter G.56

Georgia Institute of Technology and
Emory University
The Wallace H. Coulter Department of
Biomedical Engineering
Atlanta, GA, USA
charlie.kemp@bme.gatech.edu



Charles C. Kemp is an Assistant Professor in the Department of Biomedical Engineering at Georgia Tech and Emory University. He is Director of the Center for Healthcare Robotics in the Health Systems Institute. He received a Doctorate in Electrical Engineering and Computer Science from MIT in 2005. His current research focuses on autonomous robot manipulation and human-robot interaction for healthcare.

Wisama Khalil

Chapter B.14

Université de Nantes
IRCCyN, ECN
Nantes, France
wisama.khalil@irccyn.ec-nantes.fr



Wisama Khalil received the PhD and the Doctorat d'Etat degrees in Robotics and Control Engineering from Montpellier, France, in 1976 and 1978, respectively. Since 1983 he has been Professor of Robotics in the Ecole Centrale de Nantes. He carries out his research within the Robotics team of the IRCCyN (Institut de recherche en Communication et Cybernétique de Nantes-UMR-CNRS n6597). His research interest includes, kinematics, dynamics, calibration, identification, and control of robots.



Lindsay Kleeman

Chapter C.21

Monash University
Department of Electrical and Computer
Systems Engineering
Monash, VIC, Australia
kleeman@eng.monash.edu.au

Lindsay Kleeman received the bachelor degrees in Electrical Engineering and Mathematics both with university medals, and, in 1986, the PhD degree from the University of Newcastle. He is currently an Associate Professor at Monash University, Australia with over 100 publications including a best paper award at IROS99. His research interests include mobile robotics, ultrasonic sensing, and digital systems.



Tetsunori Kobayashi

Chapter G.58

Waseda University
Department of Computer Science
Tokyo, Japan
koba@waseda.jp

Tetsunori Kobayashi received the BE, ME, and PhD degrees from Waseda University in 1980, 1982, and 1985, respectively. He worked at Hosei University as a Lecturer and Associate Professor from 1985. In 1991, he joined Waseda University and has been a Professor there since 1997. His research interests include human-computer interaction using linguistic and paralinguistic information, as well as fundamentals of pattern recognition.

Kurt Konolige

SRI International
Artificial Intelligence Center
Menlo Park, CA, USA
konolige@ai.sri.com



Chapter C.22

Kurt Konolige is a Principal Computer Scientist at the Artificial Intelligence Center of SRI International, a Consulting Professor of Computer Science at Stanford University, and a Fellow of the AAAI. He received the PhD degree in Computer Science from Stanford University in 1984. His recent research has concentrated on real-time perception and navigation for mobile robots. He teaches a course in mobile robotics at Stanford University, and codeveloped the Pioneer and AmigoBot robot lines.

David Kortenkamp

TRAC Labs Inc.
Houston, TX, USA
kortenk@trac labs.com



Chapter A.8

David Kortenkamp is a Senior Scientist with TRAC Labs, Inc. supporting NASA Johnson Space Center's Automation, Robotics, and Simulation Division. He received the BS degree in Computer Science from the University of Minnesota and the MS and PhD degrees in Computer Science from the University of Michigan. His research at NASA focuses on intelligent control architectures, human-robot interaction, and autonomous execution systems.

Kazuhiro Kosuge

Tohoku University
Department of Bioengineering and Robotics
Graduate School of Engineering
Sendai, Japan
kosuge@irs.mech.tohoku.ac.jp

Chapter F.54

Professor Kazuhiro Kosuge received the BE, ME, and PhD degrees in Control Engineering from the Tokyo Institute of Technology, in 1978, 1980, and 1988, respectively. From 1980 to 1982, he was with Denso Co. Ltd. After serving as a Research Associate at Tokyo Institute of Technology and as an Associate Professor at Nagoya University, he has been a Professor at Tohoku University since 1995, and is currently with the Department of Bioengineering and Robotics.

Roman Kuc

Yale University
Department of Electrical Engineering
New Haven, CT, USA
kuc@yale.edu

Chapter C.21

Dr. Kuc is Director of Educational Affairs in Engineering and Director of the Intelligent Sensors Laboratory at Yale University. He received the BSEE degree from the Illinois Institute of Technology, Chicago, in 1968, and the PhD degree in Electrical Engineering from Columbia University, New York, in 1977. His interests cover digital signal processing with applications to bioengineering, robotics and telecommunications, inverse problems, intelligent sensors, modeling biological systems, as well as embedded systems.

James Kuffner

Carnegie Mellon University
The Robotics Institute
Pittsburgh, PA, USA
kuffner@cs.cmu.edu



Chapter D.26

James Kuffner received the PhD degree in Computer Science from Stanford University in 1999. He was a Japan Society for the Promotion of Science Postdoctoral Research Fellow at the University of Tokyo until 2001. He joined the faculty at Carnegie Mellon University in 2002 and is currently an Assistant Professor. He has published over 100 technical papers and received the Okawa Foundation Award for Young Researchers in 2007.

Vijay Kumar

University of Pennsylvania
Department of Mechanical Engineering and Applied Mechanics
Philadelphia, PA, USA
kumar@grasp.upenn.edu



Chapter E.41

Vijay Kumar is the UPS Foundation Professor and the Chairman of the Department of Mechanical Engineering and Applied Mechanics at the University of Pennsylvania. He received his PhD from the Ohio State University. He won the NSF Presidential Young Investigator Award in 1991, the Lindback Award for Distinguished Teaching in 1996, and the IEEE ICRA Kawamori Best Paper Award in 2004. He is a Fellow of the ASME and the IEEE.



Florent Lamiroux

Chapter E.35

LAAS-CNRS
Toulouse, France
florent@laas.fr

Florent Lamiroux graduated from the Ecole Polytechnique Paris in 1993. He received the PhD degree in Computer Science from the Institut National Polytechnique de Toulouse in 1997 for his research on Mobile Robots. He spent two years at Rice University as a Research Associate and is currently Chargé de Recherche at LAAS-CNRS, where he has worked for years on nonholonomic mobile robots. His interest is now on humanoid robots.



Jean-Paul Laumond

Chapter E.35

LAAS-CNRS
Toulouse, France
jpl@laas.fr

Jean-Paul Laumond, PhD 1984, IEEE Fellow and 2006–2007 Distinguished Lecturer, is Directeur de Recherche at LAAS-CNRS in Toulouse, France. His research is on robot motion planning and control. In 2001 he created Kineo CAM, a spin-off company devoted to market motion planning technology (2005 IEEE-IFR prize). He is currently co-director of the French–Japanese laboratory JRL. He has published more than 100 papers in computer science, automatic control, robotics, and neuroscience.

Steven M. LaValle

University of Illinois
Department of Computer Science
Urbana, IL, USA
lavalle@cs.uiuc.edu



Chapter A.5

Steve LaValle is an Associate Professor of Computer Science at the University of Illinois. His interests include motion planning, geometric algorithms, control theory, and sensing. He recently authored the book (free online) *Planning Algorithms*.

John J. Leonard

Massachusetts Institute of Technology
Department of Mechanical Engineering
Cambridge, MA, USA
jleonard@mit.edu



Chapter E.37

John J. Leonard is Professor of Mechanical and Ocean Engineering in the MIT Department of Mechanical Engineering. He is also a member of the MIT Computer Science and Artificial Intelligence Laboratory. He holds the degrees of BSE in Electrical Engineering and Science from the University of Pennsylvania (1987) and PhD in Engineering Science from the University of Oxford (1994). His research addresses the problems of navigation, mapping, and persistent autonomy for autonomous mobile robots.



Kevin Lynch

Chapter D.27

Northwestern University
Mechanical Engineering Department
Evanston, IL, USA
kmlynch@northwestern.edu

Kevin Lynch received the PhD degree in Robotics from Carnegie Mellon University in 1996. Since 1997 he has been faculty in the Mechanical Engineering Department of Northwestern University, where he is a McCormick Professor of Teaching Excellence. His research interests include robotic manipulation, motion planning, underactuated dynamic systems, human–robot interaction, and self-organizing systems. He received the 2001 IEEE Early Academic Career Award in Robotics and Automation.



Alan M. Lytle

Chapter F.47

National Institute of Standards and Technology
Construction Metrology and Automation Group
Gaithersburg, MD, USA
alan.lytle@nist.gov

Alan Lytle leads the Construction Metrology and Automation Group, National Institute of Standards and Technology (NIST). He received the BS degree in Electrical Engineering and the MS degree in Environmental Engineering from Virginia Tech in 1986 and 1994, respectively. Current work includes research towards automated construction, on-site component tracking, and construction object recognition using 3-D imaging systems. Awards include the 1993 SAME Bliss Medal for teaching excellence, Department of Commerce Bronze Medal, and the SPAR Achievement Award.

Maja J. Matarić

University of Southern California
Computer Science Department
Los Angeles, CA, USA
mataric@usc.edu



Chapter E.38

Maja Matarić is Professor of Computer Science and Neuroscience at the University of Southern California and Founding Director of the USC Center for Robotics and Embedded Systems. She received the PhD and MS degrees from MIT in Computer Science and AI in 1994 and 1990. She is a recipient of numerous awards and is featured in the movie “Me & Isaac Newton”. Her research into socially assistive robotics is aimed at endowing robots with the ability to help people through individual assistance and team cooperation.

Yoshio Matsumoto

Osaka University
Department of Adaptive Machine
Systems Graduate School of Engineering
Suita, Osaka, Japan
matsumoto@ams.eng.osaka-u.ac.jp



Chapter G.56

Dr. Yoshio Matsumoto received the PhD degree from the University of Tokyo in 1998. He then joined the Australian National University, where he developed a real-time vision system for gaze measurement. He moved to Nara Institute of Science and Technology in 1999, and is a Professor at Osaka University since 2007. His research interests include real-time vision, human-robot interaction, and mobile robotics.

J. Michael McCarthy

University of California at Irvine
Department of Mechanical and Aerospace
Engineering
Irvine, CA, USA
jmmccart@uci.edu

Chapter A.3

J. Michael McCarthy received the PhD degree from Stanford University in 1979, and joined the faculty at the University of California, Irvine in 1986. He is the author of two books *Introduction to Theoretical Kinematics* and *Geometric Design of Linkages*, and has served as Editor of the ASME Journal of Mechanical Design and is the Founding Editor of the ASME Journal of Mechanisms and Robotics.

Claudio Melchiorri

Università degli Studi di Bologna
Dipartimento di Elettronica Informatica e
Sistemistica
Bologna, Italy
claudio.melchiorri@unibo.it

Chapter B.15

Claudio Melchiorri is Professor of Robotics at the University of Bologna, Italy. He has served in the organization of several international conferences, and was Chair of the NOC of SYROCO 2006. He has published about 200 scientific papers, three books on digital control, and is co-editor of three books on robotics. His research interests include dexterous manipulation, haptics, telemanipulation, nonlinear control. He is a Senior Member of the IEEE.

Arianna Menciassi

Scuola Superiore Sant'Anna
CRIM Lab
Pisa, Italy
arianna@sssup.it



Chapter F.52

Arianna Menciassi received the Laurea degree in physics from the Pisa University and the PhD degree in Bioengineering from the Scuola Superiore Sant'Anna (SSSA), Pisa, Italy, in 1995 and 1999, respectively. She is an Associate Professor of Biomedical Robotics at SSSA. She also teaches at Pisa University. Her research interests include biomedical micro- and nanorobotics, microsystem technologies, and micromechatronics. She was awarded the Best Manipulation Paper Award at ICRA in 2001.

Jean-Pierre Merlet

INRIA Sophia-Antipolis
Sophia-Antipolis, France
jean-pierre.merlet@sophia.inria.fr



Chapter B.12

J.-P. Merlet is senior researcher at INRIA, where he works on interval analysis and robotics. He is the author of a book on parallel robots and of over 200 papers in robotics, mechanism theory, and numerical analysis. He is the chair of IFToMM France and has received the Altran Award and the “Micron d’Or” Award for his microrobot for surgical endoscopy.



Giorgio Metta

Chapter [G.62](#)

Italian Institute of Technology
Department of Robotics, Brain and
Cognitive Sciences
Genova, Italy
pasa@liralab.it

Giorgio Metta is Senior Scientist at the Italian Institute of Technology and Assistant Professor at the University of Genoa. His research activities are in the fields of biologically motivated and humanoid robotics developed in collaboration with leading European and international scientists from different disciplines such as neuroscience and psychology. He also worked at MIT on various humanoid robots.



Jean-Arcady Meyer

Chapter [G.60](#)

Université Pierre et Marie Curie – CNRS
Institut des Systèmes Intelligents et de
Robotique
Paris, France
jean-arcady.meyer@lip6.fr

Jean-Arcady Meyer is Emeritus Research Director at CNRS. He is trained as an Engineer, graduated in Human and Animal Psychology, and received a PhD in Biology. He is the founder of the journal Adaptive Behavior, and his primary scientific interests are in adaptive behaviors in natural and artificial systems. He is the main coordinator of the Psikharpax project.

François Michaud

Université de Sherbrooke
Department of Electrical Engineering and
Computer Engineering
Sherbrooke, Québec, Canada
francois.michaud@usherbrooke.ca



Chapter [E.38](#)

François Michaud is the Canada Research Chairholder in Mobile Robots and Autonomous Intelligent Systems at the Université de Sherbrooke, Québec, Canada. In 2003, he received the Young Engineer Achievement Award from the Canadian Council of Professional Engineers. His research interests are in architectural methodologies for intelligent decision-making, design of autonomous mobile robotics, social robotics, human-robot interaction, robot learning, and intelligent systems.

David P. Miller

University of Oklahoma
School of Aerospace and Mechanical
Engineering
Norman, OK, USA
dpmiller@ou.edu



Chapter [F.55](#)

David P. Miller is the Wilkonson Chair Professor of Intelligent Systems at the University of Oklahoma. His teaching and research span the schools of Aerospace and Mechanical Engineering, Computer Science, and Bioengineering. His primary research areas are planetary rovers, assistive technology, and robots for STEM education. He is a co-founder of the KISS Institute for Practical Robotics, which produces educational robotics programs.



Javier Minguez

Chapter [E.35](#)

Universidad de Zaragoza
Departamento de Informática e
Ingeniería de Sistemas Centro Politécnico
Superior
Zaragoza, Spain
jminguez@unizar.es

Javier Minguez received the physics degree in 1996 from the Universidad Complutense de Madrid, Spain, and the PhD degree in computer science and systems engineering in 2002 from the University of Zaragoza, Spain. He is currently a full-time Researcher in the Robotics Group at the University of Zaragoza. His research interests are mobile robots and human-machine interaction.



Pascal Morin

Chapter [E.34](#)

INRIA Sophia-Antipolis
Sophia-Antipolis, France
pascal.morin@inria.fr

Pascal Morin obtained the PhD degree from Ecole des Mines de Paris, France, in 1996, and then spent one year as a Postdoctoral Fellow at the California Institute of Technology, Pasadena, USA. Since 1997, he has been Chargé de Recherche at the French research institute INRIA. His research interests include control of nonlinear systems and its applications to mechanical systems.

Robin R. Murphy

University of South Florida
Computer Science and Engineering
Tampa, FL, USA
murphy@cse.usf.edu



Chapter F.50

Robin Roberson Murphy is director of the Institute for Safety, Security, and Rescue Technology at the University of South Florida. She received the BME degree in Mechanical Engineering, and the MS and the PhD degrees in Computer Science in 1980, 1989, and 1992, respectively, from Georgia Tech. She received the NIUSR Eagle Award for her work at the World Trade Center.

Daniele Nardi

Università degli Studi di Roma "La Sapienza"
Dipartimento di Informatica e Sistemistica "A. Ruberti"
Roma, Italy
nardi@dis.uniroma1.it



Chapter F.50

Dr. Daniele Nardi is Full Professor at the Engineering Faculty of the University "Sapienza" Rome. He received the Laurea degree in Electronic Engineering from Politecnico, Torino in 1981 and the Master degree in Computer and System Engineering from Sapienza Università di Roma in 1984. His research interests are in artificial intelligence and cognitive robotics. He is Trustee of the RoboCup Federation and Director of the Research Laboratory Intelligent Systems for Emergency and Civil Defense at the Istituto Superiore Antincendi, Roma.

Bradley J. Nelson

ETH Zentrum
Institute of Robotics and Intelligent Systems
Zürich, Switzerland
bnelson@ethz.ch

Chapter B.18

Brad Nelson received the PhD in Robotics from Carnegie Mellon University in 1995. He became Assistant Professor at the University of Illinois-Chicago in 1995, Associate Professor at the University of Minnesota in 1998, and Professor of Robotics and Intelligent Systems at ETH Zurich in 2002. His research involves microrobotics and nanorobotics.

Günter Niemeyer

Stanford University
Department of Mechanical Engineering
Design Group, Terman Engineering Center
Stanford, CA, USA
gunter.niemeyer@stanford.edu

Chapter D.31

Dr. Günter Niemeyer directs the Stanford Telerobotics Laboratory, where he studies the dynamics and control of physical human-robot interactions, in particular haptic perception and teleoperation under time delay. Applications range from telerobotic surgery to remote satellite servicing. He received the MS and PhD degrees from MIT and helped develop Intuitive Surgical's daVinci system, now in use at hundreds of hospitals worldwide.

Klas Nilsson

Lund University
Department of Computer Science
Lund, Sweden
klas@cs.lu.se



Chapter F.42

Working as an Associate Professor at the Department of Computer Science, Dr. Nilsson is involved in several research projects and teaching in real-time programming. His research includes robot control systems and related software technologies, considering implementation and industrially relevant test cases. The Laboratory for Application Languages and the Real-Time Garbage Collector developed within his group have been used for programming and real-time control of industrial robots from ABB.

Stefano Nolfi

Consiglio Nazionale delle Ricerche (CNR)
Istituto di Scienze e Tecnologie della Cognizione
Roma, Italy
stefano.nolfi@istc.cnr.it



Chapter G.61

Dr. Stefano Nolfi is Director of the Laboratory of Autonomous Robots and Artificial Life within the Institute of Cognitive Sciences and Technologies of the Italian National Research Council (CNR-ISTC). His research activities focus on evolutionary robotics, collective robotics, adaptive behavior, complex systems, and embodied cognitive sciences.



Illah R. Nourbakhsh

Chapter F.55

Carnegie Mellon University
The Robotics Institute
Pittsburgh, PA, USA
illah@cs.cmu.edu

Illah R. Nourbakhsh is an Associate Professor of Robotics and head of the Robotics Masters Program in The Robotics Institute at Carnegie Mellon University. He is co-founder of the Toy Robots Initiative, Director of the Center for Innovative Robotics, and Director of the Community Robotics, Education and Technology Empowerment Lab. He co-authored the textbook *Introduction to Autonomous Mobile Robots*.



Jonathan B. O'Brien

Chapter F.47

University of New South Wales
School of Civil and Environmental
Engineering
Sydney, Australia
j.obrien@unsw.edu.au

Jonathan O'Brien is the Founder of Construction Robotics Ltd. in the UK and a former President of the International Association for Automation and Robotics in Construction. He is Professor in Civil and Environmental Engineering at the University of NSW in Sydney, Australia. He is a pioneer of construction robotics in Australia.

Allison M. Okamura

The Johns Hopkins University
Department of Mechanical Engineering
Baltimore, MD, USA
aokamura@jhu.edu



Chapter D.30

Allison Okamura received the PhD degree from Stanford University in 2000 and is currently an Associate Professor of Mechanical Engineering at Johns Hopkins University. Her awards include the 2004 NSF CAREER Award, and the 2005 IEEE Robotics and Automation Society Early Academic Career Award. Her research interests are haptics, teleoperation, virtual environments, medical robotics, smart prosthetics, rehabilitation engineering, and engineering education.

Fiorella Operto

Scuola di Robotica
Sampierdarena, Genova, Italy
operto@scuoladirobotica.it



Chapter G.64

Dr. Operto is a scholar of the Philosophy of Science and of Science Ethics. She has acquired specific experience in science dissemination and popularization. She contributed to the birth of the scientific book series "I Dialoghi". She has cooperated with the Italian National Research Council in promoting the knowledge and understanding of robotics. In 2000 she cofounded the Association School of Robotics and is contributing to the development of roboethics.

David E. Orin

Chapter A.2

For biographical profile, please see the section "About the Part Editors" on page XXI.



Giuseppe Oriolo

Chapter B.11

Università degli Studi di Roma "La
Sapienza"
Dipartimento di Informatica e
Sistemistica "A. Ruberti"
Roma, Italy
oriolo@dis.uniroma1.it

Giuseppe Oriolo is an Associate Professor of Automatic Control at the Dipartimento di Informatica e Sistemistica of the Università di Roma "La Sapienza" and the Coordinator of the DIS Robotics Laboratory. His research interests are in the area of nonlinear control and robotics, and in particular in robot control, motion planning, trajectory planning, redundant robotic systems, control of underactuated and nonholonomic systems, sensor-based localization, navigation, and perception for mobile robots. He is a Senior Member of the IEEE.



Michel Parent

Chapter F.51

INRIA Rocquencourt
Le Chesnay, France
michel.parent@inria.fr

Michel Parent is currently the Program Manager at INRIA of the R&D team on advanced road transport (IMARA research group). This group focuses on research and development of new forms of road transport and in particular on fully automated vehicles (the cybercars). Michel Parent has an engineering degree from the French Aeronautics School (ENSAE), the Masters degree in Operation Research, and the PhD in Computer Science, both from Case Western Reserve University, Cleveland, OH.

Frank C. Park

Chapter B.10

For biographical profile, please see the section “About the Part Editors” on page XXI.

Lynne E. Parker

University of Tennessee
Department of Electrical Engineering and
Computer Science
Knoxville, TN, USA
parker@eecs.utk.edu



Chapter E.40

Dr. Parker is a Professor in the Department of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville. She received the PhD degree from the Massachusetts Institute of Technology. Her research is in the areas of distributed multi-robot systems, sensor networks, and artificial intelligence. She received the Presidential Early Career Award for Scientists and Engineers (PECASE) Award in 2000.

Michael A. Peshkin

Northwestern University
Department of Mechanical Engineering
Evanston, IL, USA
peshkin@northwestern.edu



Chapter G.57

Michael Peshkin received the doctorate in Physics from Carnegie Mellon University in 1988. His current research interests include human-machine physical interaction, human motion control, and novel sensors and actuators. He is cofounder of three companies: Mako Surgical in image guided surgery, Cobotics in industrial material handling, and Kinea Design in rehabilitation robotics, prosthetics, assistive devices, and other human-interactive applications of robotics.

J. Norberto Pires

Chapter F.42



Universidade de Coimbra
Departamento de Engenharia Mecânica
Coimbra, Portugal
norberto@robotics.dem.uc.pt

J. Norberto Pires received the PhD degree in 1999 in Robotics and Automation from the University of Coimbra, Portugal. Since 1991 he has been with the Mechanical Engineering staff of the University of Coimbra, where he currently serves as Auxiliary Professor. His research interests include force control, industrial robotics, object-oriented and distributed programming, flexible manufacturing systems, and human-machine interfaces. He is an IEEE Senior Member and Vice President of the IEEE-PT branch.

Erwin Prassler

Chapter F.54



Fachhochschule Bonn-Rhein-Sieg
Fachbereich Informatik
Sankt Augustin, Germany
erwin.prassler@fh-brs.de

Dr. Prassler received a Master's degree in Computer Science from the Technical University of Munich in 1985. In 1989, he joined the Research Institute for Applied Knowledge Processing in Ulm, where he headed a research group working in the field of mobile robots and service robotics where he received his PhD in Computer Science in 1996. In 2004, Dr. Prassler was appointed Associate Professor at the University of Applied Science Bonn-Rhein-Sieg.

Domenico Prattichizzo

Chapter D.28

Università degli Studi di Siena
Dipartimento di Ingegneria
dell'Informazione
Siena, Italy
prattichizzo@ing.unisi.it



Domenico Prattichizzo received the MS degree in Electronics Engineering and the PhD degree in Robotics and Automation from the University of Pisa, Italy in 1991 and 1995, respectively. Since 2002 he has been an Associate Professor of Robotics and Automation at the University of Siena. In 1994, he was Visiting Scientist at the Artificial Intelligence Laboratory at MIT, Cambridge. His research interests are in the area of robotic grasping, haptics, distance learning, visual servoing, geometric control, and distance learning.

Carsten Preusche

Deutsches Zentrum für Luft- und Raumfahrt (DLR) Oberpfaffenhofen
Institut für Robotik und Mechatronik
Wessling, Germany
carsten.preusche@dlr.de



Chapter D.31

Carsten Preusche became a Scientist at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) in 1999. Since 2004 he is head of the telepresence and telerobotic group. His work is mainly focused on the control of time-delayed telerobotic systems and the design of multimodal human–system interfaces.

William R. Provancher



University of Utah
Department of Mechanical Engineering
Salt Lake City, UT, USA
wil@mech.utah.edu

Chapter C.19

Dr. Provancher holds the BS degree in Mechanical Engineering and the MS degree in Materials Science both from the University of Michigan, and the PhD degree in Mechanical Engineering from Stanford University. He is currently an Assistant Professor in the Department of Mechanical Engineering at the University of Utah. His active areas of research include haptics, tactile sensing and feedback, and the design of novel climbing robots.

David J. Reinkensmeyer



University of California at Irvine
Mechanical and Aerospace Engineering
Irvine, CA, USA
dreinken@uci.edu

Chapter F.53

David Reinkensmeyer received the PhD degree in Electrical Engineering from the University of California at Berkeley in 1993, and was a Postdoctoral Fellow at the Rehabilitation Institute of Chicago. He is currently a Professor in the Department of Mechanical and Aerospace Engineering at the University of California, Irvine. His research interests are in neuromuscular control, motor learning, robotics, and neurorehabilitation.

Alfred Rizzi

Boston Dynamics
Waltham, MA, USA
arizzi@bostondynamics.com



Chapter E.39

Dr. Alfred Rizzi is the Lead Robotics Scientist at Boston Dynamics. He served as an Associate Research Professor in the Robotics Institute at Carnegie Mellon University from 1998–2006 where he directed research projects focused on hybrid sensor-based control of complex and distributed dynamical systems. These projects included the development of embedded software systems and automated behaviors for novel legged mobile robots. Dr. Rizzi received the MS and PhD degrees from Yale University in 1990 and 1994, respectively.

Jonathan Roberts

Commonwealth Scientific and Industrial
Research Organisation (CSIRO)
ICT Centre, Autonomous Systems
Laboratory
Kenmore, QLD, Australia
jonathan.roberts@csiro.au



Chapter F.49

Jonathan Roberts completed an honours degree in Aerospace Engineering and the PhD degree in Computer Vision, both at the University of Southampton, UK. In 1995, he joined CSIRO in Australia to work in the area of field robotics including UAVs, AUVs, and large AGVs for mining and metal production. In 2006 Jonathan was elected President of the Australian Robotics and Automation Association.

Daniela Rus

Chapter E.41

For biographical profile, please see the section “About the Part Editors” on page XXII.

Kamel S. Saidi



National Institute of Standards and
Technology
Building and Fire Research Laboratory
Gaithersburg, MD, USA
kamel.saidi@nist.gov

Chapter F.47

Dr. Saidi holds the PhD degree in Civil Engineering from the University of Texas at Austin (UT), where he co-invented an electric power-plant boiler maintenance robot. Dr. Saidi is currently a UT researcher and a National Institute of Standards and Technology Guest Researcher. He is developing the NIST Intelligent and Automated Construction Job Site Research Testbed and standards for 3-D imaging systems.



Claude Samson

INRIA Sophia-Antipolis
Sophia-Antipolis, France
claude.samson@inria.fr

Chapter E.34

Claude Samson is Directeur de Recherche at the French research institute INRIA. Much of his research concerns robotics and automatic control, with a particular interest in the control of nonlinear mechanical systems. He has published the book *Robot Control. The Task-Function Approach* with his colleagues Bernard Espiau and Michel Leborgne.

Stefan Schaal

University of Southern California
Computer Science and Neuroscience
Los Angeles, CA, USA
sschaal@usc.edu

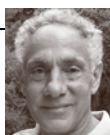


Chapter G.59

Stefan Schaal is an Associate Professor in Computer Science and Neuroscience at the University of Southern California. He has previously held positions as postdoctoral fellow at MIT, as Head of the Computational Learning Group in a Japanese ERATO project, and Adjunct Assistant Professorships at the Georgia Tech and the Pennsylvania State University. Dr. Schaal's research interests include topics of statistical and machine learning, neural networks, computational neuroscience, functional brain imaging, nonlinear dynamics and control theory, and biomimetic robotics.

Victor Scheinman

Stanford University
Department of Mechanical Engineering
Stanford, CA, USA
vds@stanford.edu



Chapter A.3

Victor Scheinman is a Consulting Professor in Mechanical Engineering at Stanford University. He has engineering degrees from MIT and Stanford. He was a founder of Automatix Inc. and Vicarm Inc. He designed and developed the Unimation PUMA robot, the Automatix/Yaskawa Robotworld, the Scheinman/Stanford Arm, and the Scheinman/MIT robot arms. He is a recipient of the ASME's Leonardo DaVinci Award for Design Engineering and the RIA's Joseph F. Engelberger Award for Robotics Technology.



James Schmiedeler

The Ohio State University
Department of Mechanical Engineering
Columbus, OH, USA
schmiedeler.2@osu.edu

Chapter A.1

James Schmiedeler is an Assistant Professor in the Department of Mechanical Engineering at The Ohio State University. His current research activities involve biped robot locomotion, human motor coordination, robot-assisted rehabilitation, human biomechanics, and design of morphing systems. His research was recognized with a National Science Foundation Presidential Early Career Award for Scientists and Engineers (PECASE) in 2007.



Roland Siegwart

ETH Zentrum
Department of Mechanical and Process
Engineering
Zürich, Switzerland
rsiegwart@ethz.ch

Chapter F.55

Roland Siegwart is Full Professor for Autonomous Systems at ETH Zürich. He was formerly Professor (1996–2006) and Vice Dean (2002–06) at EPFL, R&D Director at MECOS Traxler AG (1991–96), and held postdoc and visiting positions at Stanford University (1989/90, 2005) and NASA Ames (2006). He was Vice President for Technical Activities (2004/05) and Distinguished Lecturer (2006/08) of the IEEE Robotics and Automation Society. He is cofounder of several spin-off companies.

Reid Simmons

Carnegie Mellon University
The Robotics Institute
School of Computer Science
Pittsburgh, PA, USA
reids@cs.cmu.edu



Chapter A.8

Reid Simmons received the PhD degree in Artificial Intelligence from MIT in 1988. His research focuses on autonomous, self-reliant robots, which involves issues of robot control architectures, probabilistic planning and reasoning, robot navigation, multirobot coordination, automated assembly, and human–robot social interaction. Dr. Simmons has developed over a dozen autonomous robots, receiving the Newell Award for Research Excellence in 2004.

Dezhen Song

Texas A&M University
Department of Computer Science
College Station, TX, USA
dzsong@cs.tamu.edu



Chapter D.32

Dezhen Song received the PhD degree in Engineering from the University of California, Berkeley, in 2004. Currently, he is an Assistant Professor with Texas A&M University, College Station. His research area is networked robotics, computer vision, optimization, and stochastic modeling. Dr. Song received the Kayamori Best Paper Award of the 2005 IEEE International Conference on Robotics and Automation (with J. Yi and S. Ding) and the NSF Faculty Early Career Development (CAREER) Award in 2007.



Gaurav S. Sukhatme

University of Southern California
Department of Computer Science
Los Angeles, CA, USA
gaurav@usc.edu

Chapter E.41

Gaurav S. Sukhatme is an Associate Professor of Computer Science at the University of Southern California (USC). He earned the MS and PhD degrees in Computer Science from USC. He is the co-director of the USC Robotics Research Laboratory and the director of the USC Robotic Embedded Systems Laboratory, which he founded in 2000. His research interests are in multirobot systems and sensor/actuator networks. He is a recipient of the NSF CAREER award and the Okawa foundation research award.



Satoshi Tadokoro

Tohoku University
Graduate School of Information Sciences
Sendai, Japan
tadokoro@rm.is.tohoku.ac.jp

Chapter F.50

Satoshi Tadokoro was an Associate Professor of Kobe University in 1993–2005, and has been a Professor of Tohoku University since 2005. He established the International Rescue System Institute in 2002, where he is a President. He was a project leader of MEXT DDT Project on rescue robotics 2002–2007. He is a Trustee of The RoboCup Federation and founded RoboCupRescue.

Atsuo Takanishi

Waseda University
Department of Modern Mechanical
Engineering
Tokyo, Japan
takanisi@waseda.jp



Chapter G.58

Atsuo Takanishi is a Professor in the Department of Mechanical Engineering, Waseda University and a concurrent Professor and one of the core members of the Humanoid Robotics Institute (HRI), Waseda University. He received the MSE degree in 1982 and the PhD degree in 1988 in Mechanical Engineering from Waseda University. His current researches are related to humanoid robots and cyborgs, such as the WABIAN series (Waseda Bipedal humANoid), Waseda Leg, the Waseda Jaw, the Waseda Yamanashi, the fWaseda Flutist, the Waseda Eye, and the Waseda Talker.

Russell H. Taylor

The Johns Hopkins University
Department of Computer Science
Baltimore, MD, USA
rht@jhu.edu



Chapter F.52

Russell H. Taylor received the PhD degree in Computer Science from Stanford in 1976. He joined IBM Research in 1976, where he developed the AML robot language and managed the Computer-Assisted Surgery Group before moving in 1995 to Johns Hopkins, where he is as a Professor of Computer Science with joint appointments in Mechanical Engineering, Radiology, and Surgery. He is Director of the NSF Engineering Research Center for Computer-Integrated Surgical Systems and Technology.



Charles E. Thorpe

Carnegie Mellon University in Qatar
Qatar Office SMC 1070
Pittsburgh, PA, USA
thorpe@qatar.cmu.edu

Chapter F.51

Chuck Thorpe is Dean of the Carnegie Mellon University campus in Doha, Qatar. Prior to being appointed Dean, Thorpe was Director of the Robotics Institute at Carnegie Mellon University, where he led a research group that developed autonomous vehicles. Thorpe received the PhD degree in Computer Science from Carnegie Mellon University in 1984. In addition to serving as Dean, Thorpe is a member of the faculty, teaching Introduction to Mobile Robotics.

**Sebastian Thrun**

Chapter E.37

Stanford University
Department of Computer Science Artificial
Intelligence Laboratory
Stanford, CA, USA
thrun@stanford.edu

Sebastian Thrun is Professor of Computer Science and Electrical Engineering at Stanford University, where he directs the Stanford Artificial Intelligence Laboratory. Professor Thrun has published over 300 scientific articles and 11 books. He won numerous awards, including the 2005 DARPA Grand Challenge, the Braunschweig Research Prize, the World Technology Network Award, the Olympus Award, and numerous best paper awards. Thrun has been elected into the National Academy of Engineering, the Leopoldina, and he is an elected Fellow of the AAAI, ECCAI, and WTN.

James P. Trevelyan

Chapter F.48

The University of Western Australia
School of Mechanical Engineering
Perth, Western Australia, Australia
james.trevelyan@uwa.edu.au



James Trevelyan pioneered sheep shearing robots between 1976 and 1993, then worked on remote access laboratories, landmine clearance, and understanding engineering practice. He received the Joseph Engelberger award for contributions to robotics science in 1993, two JIRA awards, and numerous awards for distinguished teaching. He is a Fellow of Engineers Australia and an Elected Member of the IFToMM Executive Council.

Jeffrey C. Trinkle

Chapter D.28

Rensselaer Polytechnic Institute
Department of Computer Science
Troy, NY, USA
trinkle@cs.rpi.edu



Jeffrey C. Trinkle received the PhD degree from the Department of Systems Engineering at the University of Pennsylvania. Since 1987, he has held faculty positions at the University of Arizona and at Texas A&M University. From 1998 to 2003 he was at Sandia National Labs in Albuquerque, New Mexico. He is now Professor of Computer Science at Rensselaer Polytechnic Institute in Troy, New York. Professor Trinkle's primary research interests are in the areas of robotic manipulation, multibody dynamics, and automated manufacturing.

Masaru Uchiyama

Chapter D.29



Tohoku University
Department of Aerospace Engineering
Sendai, Japan
uchiyama@space.mech.tohoku.ac.jp

Masaru Uchiyama received the BE, ME, and PhD degrees from the University of Tokyo, Tokyo, Japan, in 1972, 1974, and 1977, respectively, all in Mechanical Engineering for Production. Since 1977, he has been with the School of Engineering, Tohoku University, Sendai, Japan and is currently a Professor of the Space Machines Laboratory, Department of Aerospace Engineering. His research interests include robotics, automatic control and their application to aerospace engineering.

**H.F. Machiel Van der Loos**

Chapter F.53

University of British Columbia
Department of Mechanical Engineering
Vancouver, BC, Canada
vdl@mech.ubc.ca

Dr. Machiel Van der Loos is Adjunct Professor in Mechanical Engineering at UBC, Vancouver, Canada. Until 2007, he was with the VA Rehabilitation R&D Center, Palo Alto, California, focusing on rehabilitation robotics. He is Consulting Associate Professor in Mechanical Engineering at Stanford University, collaborating on design coaching and thesis advising. He received the Ingénieur Mécanicien degree from the Swiss Federal Institute of Technology, Lausanne (1979), and an Engineer's Degree (1984) and PhD (1992) from Stanford University.

Patrick van der Smagt

Chapter G.62

Deutsches Zentrum für Luft- und
Raumfahrt (DLR) Oberpfaffenhofen
Institut für Robotik und Mechatronik
Wessling, Germany
smagt@dlr.de



Patrick van der Smagt is head of the bionics group at the Institute of Robotics and Mechatronics, German Aerospace Center (DLR). His award-winning group is inspired by biological systems and uses it to build the most advanced robotic and sensor concepts, as well as the control thereof. As Auden had Tom Rakewell say: "to follow nature as my teacher".

Gianmarco Veruggio

Consiglio Nazionale delle Ricerche
Istituto di Elettronica e di Ingegneria
dell'Informazione e delle
Telecomunicazioni
Genova, Italy
gianmarco@veruggio.it



Chapter G.64

Dr. Veruggio received the degree in Electronic Engineering from Genoa University, Italy, in 1980. He is Senior Research Scientist at IEIIT of Italian National Research Council. His research interest focuses on experimental robotics in extreme environment, and he led several underwater robotics campaigns in Antarctica and in the Arctic. In 2000 he founded the association “School of Robotics” to promote this new science among young people and the society at large. In 2002 he coined the word, and proposed the concept of Roboethics, to which he is dedicating increasing resources.

Luigi Villani

Università degli Studi di Napoli Federico II
Dipartimento di Informatica e
Sistemistica, PRISMA Lab
Napoli, Italy
luigi.villani@unina.it



Chapter A.7

Luigi Villani received the PhD degree in Electrical Engineering and Computer Science from the University of Naples in 1996, where he is Associate Professor of Automatic Control since 2002. His research interests include robot force/motion control, cooperative robots, visual servoing. He has co-authored three books and more than 100 journal and conference papers. He is Associate Editor of the IEEE Transactions on Control Systems Technology and the IEEE Transactions on Robotics.

Arto Visala

Helsinki University of Technology (TKK)
Department of Automation and Systems
Technology
Helsinki, Finland
arto.visala@tkk.fi



Chapter F.46

Arto Visala, Professor at TKK Helsinki since 2000, has an extensive experience in control engineering, automation technology, and robotics. During the last 10 years, he has been working on modelling of nonlinear dynamical systems with NN Wiener-type models, mobile multirobot systems for environment monitoring, working machine automation, machine vision systems for industry, and machine perception for mobile robots and working machines.

Kenneth Waldron

Stanford University
Department of Mechanical Engineering
Stanford, CA, USA
kwaldron@stanford.edu



Chapter A.1

Kenneth Waldron is a Professor (Research) in the Department of Mechanical Engineering of Stanford University. He has occupied academic positions at the University of New South Wales, the University of Houston, Ohio State University, and the University of Technology, Sydney, in addition to Stanford. He is President of IFToMM, the International Federation for Promotion of Mechanism and Machine Science.

Ian D. Walker

Clemson University
Department of Electrical and Computer
Engineering
Clemson, SC, USA
ianw@ces.clemson.edu



Chapter B.11

Ian D. Walker received the BS from the University of Hull, England, in 1983 and the MS and PhD degrees from the University of Texas, Austin, in 1985 and 1989, respectively. He is currently a Professor at Clemson University. His research centers on robotics, particularly novel manipulators and manipulation. He conducts research in biologically inspired “trunk, tentacle, and worm” robots.

Christian Wallraven

Max-Planck-Institut für biologische
Kybernetik
Kognitive Humanpsychophysik
Tübingen, Germany
christian.wallraven@tuebingen.mpg.de



Chapter G.63

Christian Wallraven (PhD in Physics) is a research scientist at the Max-Planck-Institute for Biological Cybernetics, Tübingen, Germany. His research addresses the fusion of cognitive psychophysics, computer graphics, and computer vision to understand fundamental perceptual processes as well as to open up new applications in the field of computer science.

**Brian Wilcox**

Chapter F.45

California Institute of Technology
Jet Propulsion Laboratory
Pasadena, CA, USA
brian.h.wilcox@jpl.nasa.gov

Brian Wilcox is at the Jet Propulsion Laboratory and is the Principal Investigator for the ATHLETE robot. He was the Supervisor of the Robotic Vehicles Group at JPL from 1985–2005, leading the development of rover technology leading to the Sojourner rover in 1997 and subsequent Mars rovers. He holds six patents and has received the NASA Exceptional Engineering Achievement Medal.

Jing Xiao

Chapter D.26

University of North Carolina
Department of Computer Science
Charlotte, NC, USA
xiao@unc.edu



Jing Xiao received the PhD degree from the University of Michigan in 1990. She is currently a Professor of Computer Science at the University of North Carolina, Charlotte. She served as the Director of the Robotics and Human Augmentation Program at the U.S. National Science Foundation from 1998 to 2000. Her research covers robotics and haptics, and she has about 100 publications.

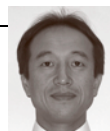
Dana R. Yoerger

Chapter F.43

Woods Hole Oceanographic Institution
Department of Applied Ocean Physics
and Engineering
Woods Hole, MA, USA
dyoerger@whoi.edu



Dana Yoerger's research interests include underwater vehicle dynamics and control, vehicle design, and autonomous scientific survey. He has contributed to the design and control systems of the Jason remotely operated vehicles, the Autonomous Benthic Explorer, and the Nereus vehicle, designed to operate to 11000 m depth. He has participated in over 50 oceanographic research cruises, and conducted 210 autonomous vehicle dives.

**Kazuhito Yokoi**

Chapter G.56

National Institute of Advanced Industrial
Science and Technology (AIST)
Intelligent Systems Research Institute
Tsukuba, Japan
kazuhito.yokoi@aist.go.jp

Kazuhito Yokoi received the PhD degree from the Tokyo Institute of Technology. He is currently a co-director of AIST-CNRS Joint Robotics Laboratory (JRL), Intelligent Systems Research Institute, National Institute of Advanced Industrial Science and Technology (AIST), at Tsukuba, Japan. He is also an Adjunct Professor of Cooperative Graduate School at University of Tsukuba. His research interests include humanoids and human-centered robotics.

**Kazuya Yoshida**

Chapter F.45

Tohoku University
Department of Aerospace Engineering
Sendai, Japan
yoshida@astro.mech.tohoku.ac.jp

Kazuya Yoshida received his Dr. Eng. from the Tokyo Institute of Technology in 1990. Since 2003 he has been Professor in the Department of Aerospace Engineering, Tohoku University, Japan. He is also serving as a Visiting Lecturer at the International Space University in Strasbourg, France, since 1998. His research activities include dynamics and control of space robots covering from free-flying robots to planetary exploration rovers.

Alexander Zelinsky

Chapter F.51

For biographical profile, please see the section “About the Part Editors” on page XXII.

Detailed Contents

List of Abbreviations	LIII
------------------------------------	------

Introduction

<i>Bruno Siciliano, Oussama Khatib</i>	1
--	---

Part A Robotics Foundations

1 Kinematics

<i>Kenneth Waldron, James Schmiedeler</i>	9
1.1 Overview	9
1.2 Position and Orientation Representation	10
1.2.1 Position and Displacement	10
1.2.2 Orientation and Rotation	10
1.2.3 Homogeneous Transformations	13
1.2.4 Screw Transformations	14
1.2.5 Matrix Exponential Parameterization	16
1.2.6 Plücker Coordinates	18
1.3 Joint Kinematics	18
1.3.1 Lower Pair Joints	19
1.3.2 Higher Pair Joints	21
1.3.3 Compound Joints	22
1.3.4 6-DOF Joint	22
1.3.5 Physical Realization	22
1.3.6 Holonomic and Nonholonomic Constraints	23
1.3.7 Generalized Coordinates	23
1.4 Geometric Representation	23
1.5 Workspace	25
1.6 Forward Kinematics	26
1.7 Inverse Kinematics	27
1.7.1 Closed-Form Solutions	27
1.7.2 Numerical Methods	28
1.8 Forward Instantaneous Kinematics	29
1.8.1 Jacobian	29
1.9 Inverse Instantaneous Kinematics	30
1.9.1 Inverse Jacobian	30
1.10 Static Wrench Transmission	30
1.11 Conclusions and Further Reading	31
References	31

2	Dynamics	
	<i>Roy Featherstone, David E. Orin</i>	35
2.1	Overview	36
2.1.1	Spatial Vector Notation	36
2.1.2	Canonical Equations	36
2.1.3	Dynamic Models of Rigid-Body Systems	36
2.1.4	Kinematic Trees	37
2.1.5	Kinematic Loops	37
2.2	Spatial Vector Notation	37
2.2.1	Motion and Force	38
2.2.2	Basis Vectors	38
2.2.3	Spatial Velocity and Force	38
2.2.4	Addition and Scalar Multiplication	39
2.2.5	Scalar Product	39
2.2.6	Coordinate Transforms	39
2.2.7	Vector Products	39
2.2.8	Differentiation	40
2.2.9	Acceleration	40
2.2.10	Spatial Momentum	40
2.2.11	Spatial Inertia	41
2.2.12	Equation of Motion	41
2.2.13	Computer Implementation	42
2.2.14	Summary	43
2.3	Canonical Equations	43
2.3.1	Joint-Space Formulation	44
2.3.2	Lagrange Formulation	44
2.3.3	Operational-Space Formulation	44
2.3.4	Impact Model	45
2.4	Dynamic Models of Rigid-Body Systems	45
2.4.1	Connectivity	45
2.4.2	Link Geometry	47
2.4.3	Link Inertias	48
2.4.4	Joint Models	48
2.4.5	Example System	48
2.5	Kinematic Trees	50
2.5.1	The Recursive Newton-Euler Algorithm	50
2.5.2	The Articulated-Body Algorithm	53
2.5.3	The Composite-Rigid-Body Algorithm	54
2.5.4	Operational-Space Inertia Matrix	55
2.6	Kinematic Loops	57
2.6.1	Formulation of Closed-Loop Algorithm	57
2.6.2	Closed-Loop Algorithm	58
2.7	Conclusions and Further Reading	60
2.7.1	Multibody Dynamics	61
2.7.2	Alternative Representations	61
2.7.3	Alternative Formulations	61
2.7.4	Efficiency	61

2.7.5	Accuracy	61
2.7.6	Software Packages	62
2.7.7	Symbolic Simplification	62
2.7.8	Algorithms for Parallel Computers	62
2.7.9	Topologically-Varying Systems	62
References	62

3 Mechanisms and Actuation

<i>Victor Scheinman, J. Michael McCarthy</i>	67
3.1	Overview	67
3.2	System Features	68
3.2.1	Work Envelope	68
3.2.2	Load Capacity	68
3.2.3	Kinematic Skeleton	69
3.3	Kinematics and Kinetics	69
3.3.1	Robot Topology	69
3.3.2	Kinematics Equations	70
3.3.3	Configuration Space	71
3.3.4	Speed Ratios	71
3.3.5	Mechanical Advantage	71
3.4	Serial Robots	72
3.4.1	Design Optimization	72
3.4.2	Speed Ratios	73
3.5	Parallel Robots	73
3.5.1	Workspace	73
3.5.2	Mechanical Advantage	74
3.5.3	Specialized Parallel Robots	74
3.6	Mechanical Structure	75
3.6.1	Links	75
3.6.2	Joints	76
3.7	Joint Mechanisms	76
3.7.1	Joint Axis Structures	76
3.7.2	Actuators	78
3.7.3	Transmissions	80
3.8	Robot Performance	82
3.8.1	Robot Speed	82
3.8.2	Robot Acceleration	83
3.8.3	Repeatability	83
3.8.4	Resolution	83
3.8.5	Accuracy	83
3.8.6	Component Life and Duty Cycle	83
3.8.7	Collisions	84
3.9	Conclusions and Further Reading	84
References	84

4	Sensing and Estimation	
	<i>Henrik I. Christensen, Gregory D. Hager</i>	87
4.1	The Perception Process	88
4.2	Sensors	90
4.3	Estimation Processes	93
4.3.1	Point Estimation	94
4.3.2	Other Approaches to Estimation	97
4.3.3	Robust Estimation Methods	98
4.3.4	Data Association Techniques	100
4.3.5	Modeling Sensors	102
4.3.6	Other Uncertainty Management Methods	103
4.4	Representations	104
4.4.1	Raw Sensor Representations	104
4.4.2	Grid-Based Representations	104
4.4.3	Discrete Feature Representations	105
4.4.4	Symbolic/Graph-Based Models	105
4.5	Conclusions and Further Readings	106
	References	106
5	Motion Planning	
	<i>Lydia E. Kavradi, Steven M. LaValle</i>	109
5.1	Motion Planning Concepts	110
5.1.1	Configuration Space	110
5.1.2	Geometric Path Planning Problem	111
5.1.3	Complexity of Motion Planning	111
5.2	Sampling-Based Planning	111
5.2.1	Multi-Query Planners: Mapping the Connectivity of $\mathcal{C}_{\text{free}}$	112
5.2.2	Single-Query Planners: Incremental Search	113
5.3	Alternative Approaches	115
5.3.1	Combinatorial Roadmaps	115
5.3.2	Roadmaps in Higher Dimensions	116
5.3.3	Potential Fields	117
5.4	Differential Constraints	118
5.4.1	Concepts and Terminology	118
5.4.2	Discretization of Constraints	119
5.4.3	Decoupled Approach	119
5.4.4	Kinodynamic Planning	120
5.5	Extensions and Variations	121
5.5.1	Closed Kinematic Chains	121
5.5.2	Manipulation Planning	122
5.5.3	Time-Varying Problems	122
5.5.4	Multiple Robots	122
5.5.5	Uncertainty in Predictability	123
5.5.6	Sensing Uncertainty	124

5.6	Advanced Issues	124
5.6.1	Topology of Configuration Spaces	124
5.6.2	Sampling Theory	125
5.6.3	Computational Algebraic Geometry Techniques	126
5.7	Conclusions and Further Reading	127
	References	128

6 Motion Control

	<i>Wankyun Chung, Li-Chen Fu, Su-Hau Hsu[†]</i>	133
6.1	Introduction to Motion Control	134
6.1.1	Dynamical Model	134
6.1.2	Control Tasks	135
6.1.3	Summary	135
6.2	Joint Space Versus Operational Space Control	135
6.2.1	Joint Space Control	135
6.2.2	Operational Space Control	136
6.3	Independent-Joint Control	137
6.3.1	Controller Design Based on the Single-Joint Model	137
6.3.2	Controller Design Based on the Multijoint Model	138
6.3.3	Summary	139
6.4	PID Control	139
6.4.1	PD Control for Regulation	139
6.4.2	PID Control for Regulation	140
6.4.3	PID Gain Tuning	140
6.5	Tracking Control	141
6.5.1	Inverse Dynamics Control	141
6.5.2	Feedback Linearization	142
6.5.3	Passivity-Based Control	143
6.5.4	Summary	143
6.6	Computed-Torque Control	143
6.6.1	Computed-Torque Control	143
6.6.2	Computed-Torque-Like Control	144
6.6.3	Summary	146
6.7	Adaptive Control	147
6.7.1	Adaptive Computed-Torque Control	147
6.7.2	Adaptive Inertia-Related Control	147
6.7.3	Adaptive Control Based on Passivity	148
6.7.4	Adaptive Control with Desired Compensation	149
6.7.5	Summary	149
6.8	Optimal and Robust Control	150
6.8.1	Quadratic Optimal Control	150
6.8.2	Nonlinear \mathcal{H}_∞ Control	151
6.8.3	Passivity-Based Design of Nonlinear \mathcal{H}_∞ Control	151
6.8.4	A Solution to Inverse Nonlinear \mathcal{H}_∞ Control	152
6.9	Digital Implementation	153
6.9.1	Reference Trajectory Generation	153
6.9.2	Z-Transform for Coding	154

6.10	Learning Control	155
6.10.1	Pure P-type Learning Control	156
6.10.2	P-Type Learning Control with a Forgetting Factor	156
6.10.3	Summary	157
	References	157

7 Force Control

	<i>Luigi Villani, Joris De Schutter</i>	161
7.1	Background	161
7.1.1	From Motion Control to Interaction Control	161
7.1.2	From Indirect Force Control to Hybrid Force/Motion Control	163
7.2	Indirect Force Control	164
7.2.1	Stiffness Control	164
7.2.2	Impedance Control	167
7.3	Interaction Tasks	171
7.3.1	Rigid Environment	171
7.3.2	Compliant Environment	173
7.3.3	Task Specification	174
7.3.4	Sensor-Based Contact Model Estimation	176
7.4	Hybrid Force/Motion Control	177
7.4.1	Acceleration-Resolved Approach	177
7.4.2	Passivity-Based Approach	179
7.4.3	Velocity-Resolved Approach	181
7.5	Conclusions and Further Reading	181
7.5.1	Indirect Force Control	182
7.5.2	Task Specification	182
7.5.3	Hybrid Force/Motion Control	182
	References	183

8 Robotic Systems Architectures and Programming

	<i>David Kortenkamp, Reid Simmons</i>	187
8.1	Overview	187
8.1.1	Special Needs of Robot Architectures	188
8.1.2	Modularity and Hierarchy	188
8.1.3	Software Development Tools	188
8.2	History	189
8.2.1	Subsumption	189
8.2.2	Layered Robot Control Architectures	190
8.3	Architectural Components	193
8.3.1	Connecting Components	193
8.3.2	Behavioral Control	195
8.3.3	Executive	196
8.3.4	Planning	199
8.4	Case Study – GRACE	200
8.5	The Art of Robot Architectures	202
8.6	Conclusions and Further Reading	203
	References	204

9 AI Reasoning Methods for Robotics

<i>Joachim Hertzberg, Raja Chatila</i>	207
9.1 Knowledge Representation and Inference	208
9.1.1 Logic	208
9.1.2 Probability Theory	210
9.2 KR Issues for Robots	212
9.2.1 Logics for High-Level Robot Control	212
9.2.2 Fuzzy Logic Approaches	213
9.2.3 Reasoning under Time Constraints	214
9.3 Action Planning	214
9.3.1 Planning Domain Descriptions	215
9.3.2 Partial-Order Plan Generation	216
9.3.3 Planning Under Uncertainty	218
9.3.4 Robot Planning	219
9.4 Robot Learning	219
9.4.1 Inductive Logic Learning	220
9.4.2 Statistical Learning and Neural Networks	220
9.4.3 Reinforcement Learning	220
9.5 Conclusions and Further Reading	221
References	222

Part B Robot Structures

10 Performance Evaluation and Design Criteria

<i>Jorge Angeles, Frank C. Park</i>	229
10.1 The Robot Design Process	229
10.2 Workspace Criteria	231
10.2.1 Reaching a Set of Goal Frames	233
10.2.2 Workspace Volume and Topology	233
10.3 Dexterity Indices	235
10.3.1 Local Dexterity for Open Chains	235
10.3.2 Dynamics-Based Local Performance Evaluation	237
10.3.3 Global Dexterity Measures	237
10.3.4 Closed-Chain Dexterity Indices	238
10.3.5 Alternative Dexterity-Like Measures	238
10.4 Other Performance Indices	238
10.4.1 Acceleration Radius	238
10.4.2 Elastostatic Performance	239
10.4.3 Elastodynamic Performance	241
References	242

11 Kinematically Redundant Manipulators

<i>Stefano Chiaverini, Giuseppe Oriolo, Ian D. Walker</i>	245
11.1 Overview	245
11.2 Task-Oriented Kinematics	247
11.2.1 Task-Space Formulation	247
11.2.2 Singularities	248

11.3	Inverse Differential Kinematics	250
11.3.1	The General Solution	250
11.3.2	Singularity Robustness	251
11.3.3	Joint Trajectory Reconstruction	254
11.4	Redundancy Resolution via Optimization	255
11.4.1	Performance Criteria	255
11.4.2	Local Optimization	256
11.4.3	Global Optimization	256
11.5	Redundancy Resolution via Task Augmentation	256
11.5.1	Extended Jacobian	256
11.5.2	Augmented Jacobian	257
11.5.3	Algorithmic Singularities	258
11.5.4	Task Priority	258
11.6	Second-Order Redundancy Resolution	259
11.7	Cyclicity	260
11.8	Hyperredundant Manipulators	261
11.8.1	Rigid-Link Hyperredundant Designs	261
11.8.2	Continuum Robot Designs	263
11.8.3	Hyperredundant Manipulator Modeling	263
11.9	Conclusion and Further Reading	265
	References	265
12	Parallel Mechanisms and Robots	
	<i>Jean-Pierre Merlet, Clément Gosselin</i>	269
12.1	Definitions	269
12.2	Type Synthesis of Parallel Mechanisms	271
12.3	Kinematics	271
12.3.1	Inverse Kinematics	271
12.3.2	Forward Kinematics	272
12.4	Velocity and Accuracy Analysis	273
12.5	Singularity Analysis	274
12.5.1	General Formulation	274
12.5.2	Parallel Singularity Analysis	275
12.6	Workspace Analysis	276
12.7	Static Analysis and Static Balancing	277
12.8	Dynamic Analysis	279
12.9	Design	279
12.10	Application Examples	280
12.11	Conclusion and Further Reading	281
	References	281
13	Robots with Flexible Elements	
	<i>Alessandro De Luca, Wayne Book</i>	287
13.1	Robots with Flexible Joints	288
13.1.1	Dynamic Modeling	289
13.1.2	Inverse Dynamics	293
13.1.3	Regulation Control	296

13.1.4	Trajectory Tracking	301
13.1.5	Further Reading	305
13.2	Robots with Flexible Links	306
13.2.1	Design Issues	306
13.2.2	Modeling of Flexible Link Arms	308
13.2.3	Control	312
13.2.4	Further Reading	316
	References	317

14 Model Identification

	<i>John Hollerbach, Wisama Khalil, Maxime Gautier</i>	321
14.1	Overview	321
14.2	Kinematic Calibration	323
14.2.1	Serial-Link Robot Manipulators	323
14.2.2	Parallel Manipulator Calibration	328
14.3	Inertial Parameter Estimation	330
14.3.1	Load Inertial Parameter Estimation	330
14.3.2	Link Inertial Parameter Estimation	332
14.3.3	Link Parameter Estimation for More Complex Structures ...	333
14.4	Identifiability and Numerical Conditioning	334
14.4.1	Identifiability	334
14.4.2	Observability	337
14.4.3	Scaling	339
14.5	Conclusions and Further Reading	341
14.5.1	Relation to Other Chapters	341
14.5.2	Further Reading	342
	References	342

15 Robot Hands

	<i>Claudio Melchiorri, Makoto Kaneko</i>	345
15.1	Basic Concepts	346
15.1.1	Anthropomorphic End-Effectors	346
15.1.2	Dexterity of a Robotic Hand	347
15.2	Design of Robot Hands	347
15.2.1	Actuator Placement and Motion Transmission	347
15.2.2	Actuation Architectures	348
15.3	Technologies for Actuation and Sensing	351
15.3.1	Actuation	351
15.3.2	Sensors	351
15.4	Modeling and Control of a Robot Hand	355
15.4.1	Dynamic Effects of Flexible Transmission Systems	355
15.4.2	Transmission Model of Tendon-Outer-Tube System	357
15.4.3	Control Through Single-Acting Actuators	358
15.4.4	Control of a Robot Hand	358
15.5	Applications and Trends	359
15.6	Conclusions and Further Reading	359
	References	359

16 Legged Robots	
<i>Shuuji Kajita, Bernard Espiau</i>	361
16.1 A Brief History	362
16.2 Analysis of Cyclic Walking	363
16.2.1 A Few Points About Hopping Robots	363
16.2.2 Stability of Passive Walking	363
16.3 Control of Biped Robots Using Forward Dynamics	366
16.3.1 Configuration Space	366
16.3.2 Dynamics	366
16.3.3 Trajectory Generation	367
16.3.4 Control	368
16.4 Biped Robots in the ZMP Scheme	371
16.4.1 Mechanisms	371
16.4.2 Zero-Moment Point (ZMP)	371
16.4.3 Computed ZMP: ZMP Calculated from Robot Motion	373
16.4.4 ZMP-Based Walking Pattern Generation	375
16.4.5 ZMP-Based Walking Control	377
16.4.6 Expansion of the ZMP Concept	377
16.5 Multilegged Robots	378
16.5.1 Analysis of Static Gait	378
16.5.2 Practical Gait Design	380
16.5.3 Dynamic Quadrupeds Inspired by Mammals	382
16.5.4 Behavior-Based Multilegged Robots	382
16.6 Other Legged Robots	383
16.6.1 Leg-Wheel Hybrid Robots	383
16.6.2 Leg-Arm Hybrid Robots	383
16.6.3 Tethered Walking Robots	384
16.6.4 Wall-Climbing Robots	384
16.7 Performance Indices	385
16.7.1 Expansion of the Stability Margin Concept	385
16.7.2 Duty Factor and Froude Number	385
16.7.3 Specific Resistance	386
16.8 Conclusions and Future Trends	386
References	387
 17 Wheeled Robots	
<i>Guy Campion, Woojin Chung</i>	391
17.1 Overview	391
17.2 Mobility of Wheeled Robots	392
17.2.1 Types of Wheels	392
17.2.2 Kinematic Constraints	394
17.2.3 Robot Configuration Variables	396
17.2.4 Restriction on Robot Mobility	396
17.2.5 Characterization of Robot Mobility	397
17.2.6 The Five Classes of Wheeled Mobile Robots	398

17.3	State-Space Models of Wheeled Mobile Robots	398
17.3.1	Posture Kinematic Models	398
17.3.2	Configuration Kinematic Models	399
17.3.3	Configuration Dynamic Models	400
17.3.4	Posture Dynamic Models	401
17.3.5	Articulated Robots	401
17.4	Structural Properties of Wheeled Robots Models	403
17.4.1	Irreducibility, Controllability, and Nonholonomy	403
17.4.2	Stabilizability	404
17.4.3	Static State-Feedback Linearizability	404
17.4.4	Dynamic State-Feedback Linearizability – Differential Flatness	404
17.5	Wheeled Robot Structures	405
17.5.1	Robots with One Wheel	405
17.5.2	Robots with Two Wheels	405
17.5.3	Robots with Three Wheels	406
17.5.4	Four Robots with Four Wheels	408
17.5.5	Special Applications of Wheeled Robots	408
17.6	Conclusions	409
	References	410

18 Micro/Nanorobots

	<i>Bradley J. Nelson, Lixin Dong, Fumihito Arai</i>	411
18.1	Overview of Micro- and Nanorobotics	411
18.2	Scaling	414
18.2.1	The Size of Things	414
18.2.2	Predominate Physics at the Micro- and Nanoscales	414
18.3	Actuation at the Micro- and Nanoscales	415
18.3.1	Electrostatics	415
18.3.2	Electromagnetics	416
18.3.3	Piezoelectrics	416
18.3.4	Other Techniques	417
18.4	Sensing at the Micro- and Nanoscales	417
18.4.1	Optical Microscopy	418
18.4.2	Electron Microscopy	418
18.4.3	Scanning Probe Microscopy	418
18.5	Fabrication	419
18.5.1	Microfabrication	419
18.5.2	Nanofabrication	421
18.6	Microassembly	422
18.6.1	Automated Microassembly Systems	422
18.6.2	Microassembly System Design	423
18.6.3	Basic Microassembly Techniques	426
18.7	Microrobotics	427
18.7.1	Microrobots	428
18.7.2	Bio-microrobotics	429

18.8	Nanorobotics	431
18.8.1	Introduction to Nanomanipulation	432
18.8.2	Nanorobotic Manipulation Systems	434
18.8.3	Nanorobotic Assembly	435
18.8.4	Nanorobotic Devices	442
18.9	Conclusions	443
	References	444

Part C Sensing and Perception

19	Force and Tactile Sensors	
	<i>Mark R. Cutkosky, Robert D. Howe, William R. Provancher</i>	455
19.1	Sensor Types	456
19.1.1	Proprioceptive and Proximity Sensing	456
19.1.2	Other Contact Sensors	458
19.1.3	Kinematic Sensors	459
19.1.4	Force and Load Sensing	459
19.1.5	Dynamic Tactile Sensors	460
19.1.6	Array Sensors	461
19.2	Tactile Information Processing	464
19.2.1	Tactile Information Flow: Means and Ends of Tactile Sensing	464
19.2.2	Solid Mechanics and Deconvolution	466
19.2.3	Curvature and Shape Information	469
19.2.4	Object and Surface Identification	469
19.2.5	Active Sensing Strategies	469
19.2.6	Dynamic Sensing and Event Detection	470
19.2.7	Integration of Thermal and Other Sensors	470
19.3	Integration Challenges	471
19.4	Conclusions and Future Developments	471
	References	471
20	Inertial Sensors, GPS, and Odometry	
	<i>Gregory Dudek, Michael Jenkin</i>	477
20.1	Odometry	477
20.2	Gyroscopic Systems	479
20.2.1	Mechanical Systems	479
20.2.2	Optical Systems	480
20.2.3	MEMS	481
20.2.4	Performance	481
20.2.5	Summary	482
20.3	Accelerometers	482
20.4	IMU Packages	483
20.5	GPS	484
20.5.1	Overview	484
20.5.2	Performance Factors	486

20.5.3	Enhanced GPS	487
20.5.4	GPS Receivers and Communications	488
20.6	GPS-IMU Integration	488
20.7	Further Reading	489
20.8	Currently Available Hardware	489
References	490

21 Sonar Sensing

<i>Lindsay Kleeman, Roman Kuc</i>	491
21.1	Sonar Principles	492
21.2	Sonar Beam Pattern	494
21.3	Speed of Sound	496
21.4	Waveforms	496
21.5	Transducer Technologies	497
21.5.1	Electrostatic	497
21.5.2	Piezoelectric	498
21.5.3	MEMS	498
21.6	Reflecting Object Models	499
21.7	Artifacts	500
21.8	TOF Ranging	501
21.9	Echo Waveform Coding	503
21.10	Echo Waveform Processing	506
21.10.1	Ranging and Wide-Bandwidth Pulses	506
21.10.2	Bearing Estimation	506
21.11	CTFM Sonar	508
21.11.1	CTFM Transmission Coding	508
21.11.2	CTFM TOF Estimation	508
21.11.3	CTFM Range Discrimination and Resolution	509
21.11.4	Comparison of CTFM and Pulse-Echo Sonar	509
21.11.5	Applications of CTFM	510
21.12	Multipulse Sonar	511
21.12.1	Interference Rejection	511
21.12.2	On-the-Fly Target Classification	511
21.13	Sonar Rings	512
21.13.1	Simple Ranging Module Rings	512
21.13.2	Advanced Rings	513
21.14	Motion Effects	513
21.14.1	Moving Observation of a Plane	513
21.14.2	Moving Observation of a Corner	514
21.14.3	Moving Observation of a Edge	514
21.14.4	The Effect of a Moving Observation on the Angle of Reception	515
21.14.5	Plane, Corner, and Edge Moving Observation Arrival Angles	515
21.15	Biomimetic Sonars	515
21.16	Conclusions	516
References	517

22 Range Sensors

<i>Robert B. Fisher, Kurt Konolige</i>	521
22.1 Range Sensing Basics	521
22.1.1 Range Images and Point Sets	521
22.1.2 Stereo Vision	523
22.1.3 Laser-Based Range Sensors	527
22.1.4 Time-of-Flight Range Sensors	528
22.1.5 Modulation Range Sensors	529
22.1.6 Triangulation Range Sensors	529
22.1.7 Example Sensors	530
22.2 Registration	530
22.2.1 3-D Feature Representations	530
22.2.2 3-D Feature Extraction	532
22.2.3 Model Matching and Multiple-View Registration	533
22.2.4 Maximum-Likelihood Registration	534
22.2.5 Multiple-Scan Registration	535
22.2.6 Relative Pose Estimation	535
22.2.7 3-D Applications	536
22.3 Navigation and Terrain Classification	537
22.3.1 Indoor Reconstruction	537
22.3.2 Urban Navigation	537
22.3.3 Rough Terrain	539
22.4 Conclusions and Further Reading	540
References	540

23 3-D Vision and Recognition

<i>Kostas Daniilidis, Jan-Olof Eklundh</i>	543
23.1 3-D Vision and Visual SLAM	544
23.1.1 Pose Estimation Solution	545
23.1.2 Triangulation	545
23.1.3 Moving Stereo	546
23.1.4 Structure from Motion (SfM)	547
23.1.5 Monocular SLAM or Multiple-View SfM	548
23.1.6 Dense Depth Maps from Stereo	549
23.2 Recognition	551
23.2.1 Approaches to Recognition	551
23.2.2 Appearance-Based Methods	552
23.2.3 Matching	555
23.2.4 Constellation-Based Methods – Recognition by Parts	557
23.2.5 Place Recognition and Terrain Classification	558
23.3 Conclusion and Further Reading	558
References	559

24 Visual Servoing and Visual Tracking

<i>François Chaumette, Seth Hutchinson</i>	563
24.1 The Basic Components of Visual Servoing	564
24.2 Image-Based Visual Servo	565
24.2.1 The Interaction Matrix	565
24.2.2 Approximating the Interaction Matrix	565
24.2.3 A Geometrical Interpretation of IBVS	567
24.2.4 Stability Analysis	568
24.2.5 IBVS with a Stereo Vision System	570
24.2.6 IBVS with Cylindrical Coordinates of Image Points	570
24.2.7 IBVS with Other Geometrical Features	571
24.2.8 Direct Estimation	571
24.3 Position-Based Visual Servo	572
24.4 Advanced Approaches	574
24.4.1 Hybrid VS	574
24.4.2 Partitioned VS	576
24.5 Performance Optimization and Planning	577
24.5.1 Optimal Control and Redundancy Framework	577
24.5.2 Switching Schemes	577
24.5.3 Feature Trajectory Planning	578
24.6 Estimation of 3-D Parameters	578
24.7 Target Tracking	579
24.8 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space	580
24.9 Conclusions	581
References	582

25 Multisensor Data Fusion

<i>Hugh Durrant-Whyte, Thomas C. Henderson</i>	585
25.1 Multisensor Data Fusion Methods	585
25.1.1 Bayes' Rule	586
25.1.2 Probabilistic Grids	589
25.1.3 The Kalman Filter	590
25.1.4 Sequential Monte Carlo Methods	594
25.1.5 Alternatives to Probability	596
25.2 Multisensor Fusion Architectures	598
25.2.1 Architectural Taxonomy	598
25.2.2 Centralized, Local Interaction, and Hierarchical	600
25.2.3 Decentralized, Global Interaction, and Heterarchical	600
25.2.4 Decentralized, Local Interaction, and Hierarchical	601
25.2.5 Decentralized, Local Interaction, and Heterarchical	602
25.3 Applications	603
25.3.1 Dynamic System Control	603
25.3.2 ANSER II: Decentralized Data Fusion	604
25.4 Conclusions	608
References	608

Part D Manipulation and Interfaces

26 Motion for Manipulation Tasks 615

Oliver Brock, James Kuffner, Jing Xiao 615

 26.1 Overview 615

 26.2 Task-Level Control 618

 26.2.1 Operational Space Control 618

 26.2.2 Combined Force and Position Control 619

 26.2.3 Operational Space Control of Redundant Mechanisms 619

 26.2.4 Combining Mobility and Manipulation 620

 26.2.5 Combining Multiple Task Behaviors 620

 26.3 Manipulation Planning 622

 26.3.1 Configuration Space Formalism 622

 26.3.2 Example of a Three-DOF Planar Manipulator 625

 26.3.3 Inverse Kinematics Considerations 626

 26.4 Assembly Motion 628

 26.4.1 Topological Contact States 629

 26.4.2 Passive Compliance 629

 26.4.3 Active Compliant Motion 631

 26.5 Unifying Feedback Control and Planning 634

 26.5.1 Feedback Motion Planning 635

 26.5.2 Augmenting Global Plans with Feedback 636

 26.6 Conclusions and Further Reading 637

 References 639

27 Contact Modeling and Manipulation 647

Imin Kao, Kevin Lynch, Joel W. Burdick 647

 27.1 Overview 647

 27.1.1 Choosing a Contact Model 648

 27.1.2 Grasp/Manipulation Analysis 648

 27.2 Kinematics of Rigid-Body Contact 648

 27.2.1 Contact Constraints 649

 27.2.2 Collections of Parts 650

 27.2.3 Graphical Planar Methods 651

 27.3 Forces and Friction 652

 27.3.1 Graphical Planar Methods 653

 27.3.2 Duality of Contact Wrenches and Twist Freedoms 653

 27.4 Rigid-Body Mechanics with Friction 655

 27.4.1 Complementarity 655

 27.4.2 Quasistatic Assumption 656

 27.4.3 Examples 656

 27.5 Pushing Manipulation 658

 27.6 Contact Interfaces and Modeling 659

 27.6.1 Modeling of Contact Interface 659

 27.6.2 Pressure Distribution at Contacts 660

27.7	Friction Limit Surface	661
27.7.1	The Friction Limit Surface at a Soft Contact Interface	662
27.7.2	Example of Constructing a Friction Limit Surface	663
27.8	Contacts in Grasping and Fixture Designs	664
27.8.1	Contact Stiffness of Soft Fingers	664
27.8.2	Application of Soft Contact Theory to Fixture Design	665
27.9	Conclusions and Further Reading	666
	References	666

28 Grasping

	<i>Domenico Prattichizzo, Jeffrey C. Trinkle</i>	671
28.1	Background	671
28.2	Models and Definitions	672
28.2.1	Velocity Kinematics	673
28.2.2	Dynamics and Equilibrium	676
28.3	Controllable Twists and Wrenches	677
28.3.1	Grasp Classifications	678
28.3.2	Limitations of Rigid-Body Assumption	679
28.3.3	Desirable Properties	679
28.4	Restraint Analysis	680
28.4.1	Form Closure	681
28.4.2	Force Closure	684
28.5	Examples	687
28.5.1	Example 1: Grasped Sphere	687
28.5.2	Example 2: Grasped Polygon in the Plane	691
28.5.3	Example 3: Hyperstatic Grasps	693
28.5.4	Example 4: Duality	696
28.5.5	Example 5: Form Closure	696
28.6	Conclusion and Further Reading	697
	References	698

29 Cooperative Manipulators

	<i>Fabrizio Caccavale, Masaru Uchiyama</i>	701
29.1	A Historical Overview	701
29.2	Kinematics and Statics	703
29.2.1	Symmetric Formulation	704
29.2.2	Multifingered Manipulation	706
29.3	Cooperative Task Space	707
29.4	Dynamics and Load Distribution	708
29.4.1	Reduced-Order Models	708
29.4.2	Load Distribution	709
29.5	Task-Space Analysis	710
29.6	Control	711
29.6.1	Hybrid Control	711
29.6.2	PD Force/Motion Control	712

29.6.3	Feedback Linearization Approaches	713
29.6.4	Impedance Control	713
29.7	Conclusions and Further Reading	715
References	716

30 Haptics

<i>Blake Hannaford, Allison M. Okamura</i>	719
30.1	Overview	719
30.1.1	Human Haptics	721
30.1.2	Application Examples	722
30.2	Haptic Device Design	724
30.2.1	Mechanisms	724
30.2.2	Sensing	725
30.2.3	Actuation and Transmission	726
30.2.4	An Example Device	726
30.3	Haptic Rendering	727
30.3.1	Rendering Complex Environments	729
30.3.2	Virtual Coupling	729
30.4	Control and Stability of Haptic Interfaces	730
30.5	Tactile Display	731
30.5.1	Vibration Feedback	731
30.5.2	Contact Location, Slip, and Shear Display	732
30.5.3	Local Shape	733
30.5.4	Temperature	734
30.6	Conclusions and Further Reading	735
References	735

31 Telerobotics

<i>Günter Niemeyer, Carsten Preusche, Gerd Hirzinger</i>	741
31.1	Overview	741
31.2	Telerobotic Systems and Applications	743
31.2.1	Historical Perspective	743
31.2.2	Applications	744
31.3	Control Architectures	746
31.3.1	Supervisory Control	746
31.3.2	Shared Control	748
31.3.3	Direct and Bilateral Teleoperation	749
31.4	Bilateral Control and Force Feedback	751
31.4.1	Position/Force Control	751
31.4.2	Passivity and Stability	752
31.4.3	Transparency and Multichannel Feedback	753
31.4.4	Time Delay and Scattering Theory	754
31.4.5	Wave Variables	754
31.5	Conclusions and Further Reading	754
References	755

32 Networked Telerobots

<i>Dezhen Song, Ken Goldberg, Nak Young Chong</i>	759
32.1 Overview and Background	759
32.2 A Brief History	760
32.3 Communications and Networking	761
32.3.1 The Internet	762
32.3.2 Wired Communication Links	763
32.3.3 Wireless Links	763
32.3.4 Properties of Networked Telerobotics	764
32.3.5 Building a Networked Telerobotic System	764
32.3.6 State-Command Presentation	765
32.3.7 Command Execution/State Generation	767
32.3.8 Collaborative Control	768
32.4 Conclusion and Future Directions	769
References	769

33 Exoskeletons for Human Performance Augmentation

<i>Homayoon Kazerooni</i>	773
33.1 Survey of Exoskeleton Systems	773
33.2 Upper-Extremity Exoskeleton	775
33.3 Intelligent Assist Device	776
33.4 Control Architecture for Upper-Extremity Exoskeleton Augmentation	778
33.5 Applications of Intelligent Assist Device	780
33.6 Lower-Extremity Exoskeleton	780
33.7 The Control Scheme of an Exoskeleton	782
33.8 Highlights of the Lower-Extremity Design	786
33.9 Field-Ready Exoskeleton Systems	790
33.9.1 The ExoHiker Exoskeleton	790
33.9.2 The ExoClimber Exoskeleton	790
33.10 Conclusion and Further Reading	792
References	792

Part E Mobile and Distributed Robotics**34 Motion Control of Wheeled Mobile Robots**

<i>Pascal Morin, Claude Samson</i>	799
34.1 Background	800
34.1.1 Path Following	800
34.1.2 Stabilization of Trajectories	800
34.1.3 Stabilization of Fixed Postures	801
34.2 Control Models	801
34.2.1 Kinematics Versus Dynamics	801
34.2.2 Modeling in a Frénet Frame	803
34.3 Adaptation of Control Methods for Holonomic Systems	804
34.3.1 Stabilization of Trajectories for a Nonconstrained Point ...	804
34.3.2 Path Following with No Orientation Control	805

34.4	Methods Specific to Nonholonomic Systems	806
34.4.1	Transformation of Kinematic Models into the Chained Form	807
34.4.2	Tracking of a Reference Vehicle with the Same Kinematics	808
34.4.3	Path Following with Orientation Control	811
34.4.4	Asymptotic Stabilization of Fixed Postures	813
34.4.5	Limitations Inherent to the Control of Nonholonomic Systems	818
34.4.6	Practical Stabilization of Arbitrary Trajectories Based on the Transverse Function Approach	820
34.5	Complementary Issues and Bibliographical Guide	823
	References	825

35 Motion Planning and Obstacle Avoidance

	<i>Javier Minguez, Florent Lamiriaux, Jean-Paul Laumond</i>	827
35.1	Nonholonomic Mobile Robots: Where Motion Planning Meets Control Theory	828
35.2	Kinematic Constraints and Controllability	829
35.2.1	Definitions	829
35.2.2	Controllability	829
35.2.3	Example: The Differentially Driven Mobile Robot	830
35.3	Motion Planning and Small-Time Controllability	830
35.3.1	The Decision Problem	830
35.3.2	The Complete Problem	831
35.4	Local Steering Methods and Small-Time Controllability	832
35.4.1	Local Steering Methods that Account for Small-Time Controllability	832
35.4.2	Equivalence Between Chained-Formed and Feedback-Linearizable Systems	834
35.5	Robots and Trailers	835
35.5.1	Differentially Driven Mobile Robots	835
35.5.2	Differentially Driven Mobile Robots Towing One Trailer	835
35.5.3	Car-Like Mobile Robots	835
35.5.4	Bi-steerable Mobile Robots	835
35.5.5	Differentially Driven Mobile Robots Towing Trailers	836
35.5.6	Open Problems	836
35.6	Approximate Methods	837
35.6.1	Forward Dynamic Programming	837
35.6.2	Discretization of the Input Space	837
35.6.3	Input-Based Rapidly Exploring Random Trees	837
35.7	From Motion Planning to Obstacle Avoidance	837
35.8	Definition of Obstacle Avoidance	838
35.9	Obstacle Avoidance Techniques	839
35.9.1	Potential Field Methods (PFM)	839
35.9.2	Vector Field Histogram (VFH)	840
35.9.3	The Obstacle Restriction Method (ORM)	841
35.9.4	Dynamic Window Approach (DWA)	842

35.9.5	Velocity Obstacles (VO)	843
35.9.6	Nearness Diagram Navigation (ND)	843
35.10	Robot Shape, Kinematics, and Dynamics in Obstacle Avoidance	845
35.10.1	Techniques that Abstract the Vehicle Aspects	845
35.10.2	Techniques of Decomposition in Subproblems	846
35.11	Integration Planning – Reaction	847
35.11.1	Path Deformation Systems	847
35.11.2	Tactical Planning Systems	848
35.12	Conclusions, Future Directions, and Further Reading	849
	References	850

36 World Modeling

	<i>Wolfram Burgard, Martial Hebert</i>	853
36.1	Historical Overview	854
36.2	World Models for Indoors and Structured Environments	855
36.2.1	Occupancy Grids	855
36.2.2	Line Maps	857
36.2.3	Topological Maps	858
36.2.4	Landmark-Based Maps	859
36.3	World and Terrain Models for Natural Environments	859
36.3.1	Elevation Grids	859
36.3.2	3-D Grids and Point Sets	861
36.3.3	Meshes	862
36.3.4	Cost Maps	863
36.3.5	Semantic Attributes	864
36.3.6	Heterogeneous and Hierarchical Models	866
36.4	Dynamic Environments	866
36.4.1	Conclusion and Further Reading	867
	References	867

37 Simultaneous Localization and Mapping

	<i>Sebastian Thrun, John J. Leonard</i>	871
37.1	Overview	871
37.2	SLAM: Problem Definition	872
37.2.1	Mathematical Basis	872
37.2.2	Example: SLAM in Landmark Worlds	873
37.2.3	Taxonomy of the SLAM Problem	873
37.3	The Three Main SLAM Paradigms	875
37.3.1	Extended Kalman Filters	875
37.3.2	Graph-Based Optimization Techniques	877
37.3.3	Particle Methods	881
37.3.4	Relation of Paradigms	883
37.4	Conclusion and Future Challenges	885
37.5	Suggestions for Further Reading	886
	References	886

38 Behavior-Based Systems

<i>Maja J. Matarić, François Michaud</i>	891
38.1 Robot Control Approaches	891
38.1.1 Deliberative – Think, Then Act	892
38.1.2 Reactive – Don't Think, (Re)Act	892
38.1.3 Hybrid – Think and Act Concurrently	893
38.1.4 Behavior-Based Control – Think the Way You Act	893
38.2 Basic Principles of Behavior-Based Systems	894
38.2.1 Misconceptions	896
38.3 Basis Behaviors	897
38.4 Representation in Behavior-Based Systems	897
38.5 Learning in Behavior-Based Systems	898
38.5.1 Reinforcement Learning in Behavior-Based Systems	899
38.5.2 Learning Behavior Networks	899
38.5.3 Learning from History of Behavior Use	900
38.6 Continuing Work	902
38.6.1 Motivated Configuration of Behaviors	903
38.7 Conclusions and Further Reading	905
References	906

39 Distributed and Cellular Robots

<i>Zack Butler, Alfred Rizzi</i>	911
39.1 Modularity for Locomotion	911
39.1.1 Self-Reconfigurable Robot Locomotion	912
39.1.2 Physically Cooperative Mobile Robots	914
39.2 Modularity for Manipulation	914
39.2.1 Independent Manipulators	914
39.2.2 Reconfigurable Manipulators	915
39.3 Modularity for Geometric Reconfiguration of Robot Systems	915
39.3.1 Manually Reconfigured Systems	916
39.3.2 Shape Generation via Self-Reconfiguring Systems	916
39.3.3 Configuration Optimization	917
39.3.4 Self-Replicating Systems	918
39.4 Modularity for Robustness	918
39.5 Conclusions and Further Reading	918
References	919

40 Multiple Mobile Robot Systems

<i>Lynne E. Parker</i>	921
40.1 History	922
40.2 Architectures for Multirobot Systems	922
40.2.1 The Nerd Herd	923
40.2.2 The ALLIANCE Architecture	923
40.2.3 The Distributed Robot Architecture	924
40.3 Communication	925
40.4 Swarm Robots	926
40.5 Heterogeneity	928

40.6	Task Allocation	930
40.6.1	Taxonomy for Task Allocation	930
40.6.2	Representative Approaches	930
40.7	Learning	932
40.8	Applications	933
40.8.1	Foraging and Coverage	933
40.8.2	Flocking and Formations	933
40.8.3	Box Pushing and Cooperative Manipulation	934
40.8.4	Multitarget Observation	934
40.8.5	Traffic Control and Multirobot Path Planning	934
40.8.6	Soccer	935
40.9	Conclusions and Further Reading	935
	References	936

41 Networked Robots

	<i>Vijay Kumar, Daniela Rus, Gaurav S. Sukhatme</i>	943
41.1	Overview	943
41.2	State of the Art and Potential	945
41.3	Research Challenges	947
41.4	Control	949
41.5	Communication for Control	950
41.6	Communication for Perception	951
41.7	Control for Perception	952
41.8	Control for Communication	953
41.9	Conclusions and Further Reading	955
	References	955

Part F Field and Service Robotics

42 Industrial Robotics

	<i>Martin Hägele, Klas Nilsson, J. Norberto Pires</i>	963
42.1	A Short History of Industrial Robots	964
42.2	Typical Applications and Robot Configurations	969
42.2.1	Welding	969
42.2.2	Car Body Assembly	969
42.2.3	Painting	971
42.2.4	Material Transfer Automation	971
42.2.5	Machining	974
42.2.6	Human–Robot Cooperation for Handling Tasks	974
42.3	Kinematics and Mechanisms	975
42.4	Task Descriptions – Teaching and Programming	976
42.5	End-Effectors and System Integration	980
42.6	Conclusions and Long-Term Challenges	983
	References	985

43 Underwater Robotics
Gianluca Antonelli, Thor I. Fossen, Dana R. Yoerger 987

43.1 The Expanding Role of Marine Robotics in Oceanic Engineering 987

43.1.1 Historical Background 989

43.2 Underwater Robotics 989

43.2.1 Modeling 989

43.2.2 Sensor Systems 995

43.2.3 Actuating Systems 996

43.2.4 Mission Control System 998

43.2.5 Guidance and Control 998

43.2.6 Localization 1000

43.2.7 Underwater Manipulation 1001

43.2.8 Fault Detection/Tolerance 1002

43.2.9 Multiple Underwater Vehicles 1003

43.3 Applications 1003

43.4 Conclusions and Further Reading 1005

References 1005

44 Aerial Robotics
Eric Feron, Eric N. Johnson 1009

44.1 Background 1010

44.2 History of Aerial Robotics 1010

44.3 Applications of Aerial Robotics 1012

44.3.1 Possible Applications of Aerial Robots 1012

44.3.2 Current Applications 1013

44.4 Current Challenges 1014

44.4.1 Regulations and Certification 1014

44.4.2 Human–Machine Interfaces 1014

44.4.3 Navigation 1015

44.4.4 Agile Flight and Fault Tolerance 1015

44.4.5 Obstacle Avoidance 1015

44.4.6 Aerial Robot Landing and Interaction with Other Vehicles 1015

44.4.7 Multivehicle Coordination 1015

44.5 Basic Aerial Robot Flight Concepts 1015

44.5.1 Aerial Robot Flight and the Importance of Scales 1015

44.5.2 Propulsion Systems 1017

44.5.3 Flight Vehicle Types and Flight Regimes 1018

44.5.4 Lighter–Than–Air Systems 1019

44.6 The Entry Level for Aerial Robotics: Inner–Loop Control 1020

44.6.1 Sensing and Estimation 1020

44.6.2 Estimator Design 1021

44.6.3 Inner–Loop Control 1021

44.7 Active Research Areas 1022

44.7.1 Interfacing with the Human Infrastructure: Meeting the Regulations 1022

44.7.2 High–Agility Flight 1023

44.7.3 Take–Off, Landing, and Interaction with Other Vehicles ... 1023

44.7.4	Reactive Flight in Cluttered Environments and Obstacle Avoidance	1025
44.7.5	Path Planning and Higher-Level Planning Capabilities	1025
44.7.6	Integrated Aerial Robotic Operations: Aerial Robotics Contests	1025
44.8	Conclusions and Further Reading	1026
References	1027

45 Space Robots and Systems

	<i>Kazuya Yoshida, Brian Wilcox</i>	1031
45.1	Historical Developments and Advances of Orbital Robotic Systems ..	1032
45.1.1	Space Shuttle Remote Manipulator System	1032
45.1.2	ISS-Mounted Manipulator Systems	1033
45.1.3	ROTEX	1035
45.1.4	ETS-VII	1035
45.1.5	Ranger	1036
45.1.6	Orbital Express	1036
45.2	Historical Developments and Advances of Surface Robotic Systems .	1037
45.2.1	Teleoperated Rovers	1037
45.2.2	Autonomous Rovers	1039
45.2.3	Research Systems	1042
45.2.4	Sensing and Perception	1043
45.2.5	Estimation	1043
45.2.6	Manipulators for In Situ Science	1044
45.3	Mathematical Modeling	1044
45.3.1	Space Robot as an Articulated-Body System	1044
45.3.2	Equations for Free-Floating Manipulator Systems	1045
45.3.3	Generalized Jacobian and Inertia Matrices	1045
45.3.4	Linear and Angular Momenta	1047
45.3.5	Virtual Manipulator	1047
45.3.6	Dynamic Singularity	1047
45.3.7	Reaction Null Space (RNS)	1048
45.3.8	Equations for Flexible-Based Manipulator Systems	1048
45.3.9	Advanced Control for Flexible Structure Based Manipulators	1049
45.3.10	Contact Dynamics and Impedance Control	1050
45.3.11	Dynamics of Mobile Robots	1052
45.3.12	Wheel Traction Mechanics	1053
45.4	Future Directions of Orbital and Surface Robotic Systems	1056
45.4.1	Robotic Maintenance and Service Missions	1056
45.4.2	Robonaut	1057
45.4.3	Aerial Platforms	1057
45.4.4	Subsurface Platforms	1059
45.5	Conclusions and Further Reading	1060
References	1060

46 Robotics in Agriculture and Forestry
John Billingsley, Arto Visala, Mark Dunn 1065

46.1 Definitions 1066

46.1.1 Technological Developments 1066

46.2 Forestry 1066

46.2.1 Overview 1066

46.2.2 Robot Locomotion in Forestry 1067

46.2.3 Forestry Automation 1068

46.2.4 Machine Perception and SLAM in the Forest Environment . 1068

46.2.5 Autonomy, Teleoperation, and Fleet Management 1069

46.2.6 Autonomous Robots for Silviculture and Treatment 1069

46.2.7 Forestry Conclusions 1070

46.3 Broad Acre Applications 1070

46.3.1 An Overview of Automatic Guidance 1070

46.3.2 Sowing, Weeding, Spraying, and Broad-Acre Harvesting .. 1071

46.4 Horticulture 1071

46.4.1 Picking of Fruit and Vegetables 1071

46.4.2 Color Sorting and Produce Grading 1072

46.5 Livestock 1072

46.5.1 Robot Milking 1072

46.5.2 Sheep Shearing 1073

46.5.3 Slaughtering 1073

46.5.4 Livestock Inspection 1074

46.5.5 Robotic *Animals* 1074

46.6 Unmanned Vehicles 1075

46.7 Conclusions and Future Directions 1075

References 1075

47 Robotics in Construction
Kamel S. Saidi, Jonathan B. O'Brien, Alan M. Lytle 1079

47.1 Overview 1080

47.1.1 Industry Description 1080

47.1.2 Automation in Construction 1081

47.1.3 Robotics in Construction 1082

47.1.4 Sensing Systems in Construction 1083

47.2 Economic Aspects 1085

47.2.1 Scope 1085

47.2.2 Motivation 1085

47.2.3 Barriers 1085

47.3 Applications 1086

47.3.1 Construction 1086

47.3.2 Maintenance and Decommissioning 1091

47.4 Currently Unsolved Technical Problems 1093

47.4.1 Interoperability 1093

47.4.2 Structural Connection Systems 1093

47.4.3 Tolerances 1094

47.4.4 Power and Communications in the Field 1095

47.5	Future Directions	1095
47.6	Conclusions and Further Reading	1096
	References	1096

48 Robotics in Hazardous Applications

	<i>James P. Trevelyan, Sung-Chul Kang, William R. Hamel</i>	1101
48.1	Operation in Hazardous Environments: The Need for a Robotics Solution	1101
48.2	Applications	1103
48.2.1	Eradicating Landmines	1103
48.2.2	Hazardous Materials Handling and Operations	1110
48.3	Enabling Technologies	1114
48.3.1	Mobility Issues	1114
48.3.2	Manipulator Design and Control for Hazardous Object Handling	1117
48.3.3	Control for Hazardous Tasks	1117
48.3.4	Data Communications	1120
48.3.5	Energetics	1120
48.3.6	System Architectures for Real-Time Mission Control	1121
48.4	Conclusions and Further Reading	1122
	References	1123

49 Mining Robotics

	<i>Peter Corke, Jonathan Roberts, Jock Cunningham, David Hainsworth</i>	1127
49.1	Background	1127
49.1.1	Robotics in Mining	1129
49.2	Metalliferous Mining	1132
49.2.1	Underground Hard-Rock Metalliferous	1132
49.2.2	Scope for Robotics	1135
49.2.3	Case Study: Nonconventional Robotic Miner	1139
49.3	Underground Coal Mining	1140
49.3.1	Coal Mining Process	1141
49.3.2	Scope for Robotics	1142
49.3.3	Rapid Roadway Development	1143
49.4	Surface Coal Mining	1144
49.4.1	Dragline Excavator	1144
49.4.2	Truck-Shovel Operation	1146
49.4.3	Surface Haulage	1147
49.5	Conclusions and Further Reading	1148
	References	1148

50 Search and Rescue Robotics
Robin R. Murphy, Satoshi Tadokoro, Daniele Nardi, Adam Jacoff, Paolo Fiorini, Howie Choset, Aydan M. Erkmen 1151

50.1 Overview 1152

50.1.1 Motivation 1152

50.1.2 Rescue Robot Tasks 1152

50.1.3 Types of Rescue Robots 1153

50.2 Disaster Characteristics and Impact on Robots 1153

50.2.1 Categories and Phases of Disasters 1154

50.2.2 Natural Disasters 1155

50.2.3 Suitable Robot Technologies for Natural Disasters 1155

50.2.4 Manmade Disasters 1156

50.2.5 Suitable Robot Technologies for Manmade Disasters 1156

50.3 Robots Actually Used at Disasters 1156

50.3.1 2001 World Trade Center, United States 1157

50.3.2 2005 La Conchita Mudslide, United States 1158

50.3.3 2005 Hurricanes Katrina, Rita and Wilma, United States ... 1158

50.3.4 2006 Sago Mine Disaster, United States 1159

50.3.5 Post-Disaster Experimentation 1160

50.3.6 Search and Recovery 1161

50.4 Promising Robots and Concepts 1161

50.4.1 Alternative Ground Rescue Robot Designs 1162

50.4.2 Aerial Rescue Robots 1163

50.4.3 Unique Concepts of Operations 1164

50.5 Evaluation and Benchmarks 1165

50.5.1 Computer Simulations for Rescue Robotics 1165

50.5.2 Physical Test Beds 1166

50.5.3 Standards Activity 1167

50.6 Fundamental Problems and Open Issues 1167

50.6.1 Mobility 1168

50.6.2 Communications 1168

50.6.3 Control 1169

50.6.4 Sensors 1169

50.6.5 Power 1169

50.6.6 Human–Robot Interaction 1169

50.6.7 Evaluation 1171

50.7 Conclusions and Further Reading 1171

References 1171

51 Intelligent Vehicles
Alberto Broggi, Alexander Zelinsky, Michel Parent, Charles E. Thorpe 1175

51.1 Why Intelligent Vehicles? 1175

51.1.1 Brief History 1176

51.1.2 Benefits of Intelligent Vehicles 1177

51.2 Enabling Technologies 1178

51.2.1 Environment State 1179

51.2.2 Driver State 1180

51.2.3	Communication	1180
51.2.4	Digital Maps and Satellite Data	1180
51.3	Road Scene Understanding	1181
51.3.1	Road/Lane Tracking	1182
51.3.2	Road Sign Detection	1182
51.3.3	Traffic-Light Detection	1183
51.3.4	Visibility Assessment	1183
51.3.5	Vehicle Detection	1183
51.3.6	Pedestrian Detection	1184
51.4	Advanced Driver Assistance	1184
51.4.1	Collision Avoidance and Mitigation	1185
51.4.2	Adaptive Cruise Control	1187
51.4.3	Stop and Go	1187
51.4.4	Parking Assist	1187
51.4.5	Lane Keeping	1188
51.4.6	Lane Changing	1188
51.5	Driver Monitoring	1188
51.5.1	Driver Fatigue, Inattention, and Impairment	1189
51.5.2	Driver and Passenger Protection	1190
51.5.3	Emergency Assistance	1190
51.6	Automated Vehicles	1191
51.6.1	Operating Safely	1191
51.6.2	Traffic Congestion	1191
51.6.3	Environmental Factors	1191
51.6.4	The Automobile of the Future	1192
51.6.5	Automated Vehicle Deployment	1193
51.7	Future Trends and Prospects	1194
51.8	Conclusions and Further Reading	1195
	References	1195

52 Medical Robotics and Computer-Integrated Surgery

	<i>Russell H. Taylor, Arianna Menciassi, Gabor Fichtinger, Paolo Dario</i>	1199
52.1	Core Concepts	1200
52.1.1	Medical Robotics, Computer-Integrated Surgery, and Closed-Loop Interventions	1200
52.1.2	Factors Affecting the Acceptance of Medical Robots	1200
52.1.3	Medical Robotics System Paradigms: Surgical CAD/CAM and Surgical Assistance	1202
52.2	Technology	1204
52.2.1	Mechanical Design Considerations	1204
52.2.2	Control Paradigms	1205
52.2.3	Virtual Fixtures and Human-Machine Cooperative Systems	1206
52.2.4	Safety and Sterility	1207
52.2.5	Imaging and Modeling of Patients	1208
52.2.6	Registration	1208

- 52.3 Systems, Research Areas, and Applications 1209
 - 52.3.1 Nonrobotic Computer-Assisted Surgery: Navigation and Image Overlay Devices 1209
 - 52.3.2 Orthopaedic Systems 1209
 - 52.3.3 Percutaneous Needle Placement Systems 1210
 - 52.3.4 Telesurgical Systems 1212
 - 52.3.5 Microsurgery Systems 1213
 - 52.3.6 Endoluminal Robots 1213
 - 52.3.7 Sensorized Instruments and Haptic Feedback 1214
 - 52.3.8 Surgical Simulators and Telerobotic Systems for Training .. 1215
 - 52.3.9 Other Applications and Research Areas 1216
- 52.4 Conclusion and Future Directions 1217
- References** 1218

53 Rehabilitation and Health Care Robotics

- H.F. Machiel Van der Loos, David J. Reinkensmeyer* 1223
- 53.1 Overview 1223
 - 53.1.1 Taxonomy of Rehabilitation Robotics 1224
 - 53.1.2 World Demographics 1224
 - 53.1.3 Short History of the Field of Rehabilitation Robotics 1225
- 53.2 Physical Therapy and Training Robots 1227
 - 53.2.1 Grand Challenges and Roadblocks 1227
 - 53.2.2 Movement Therapy after Neurologic Injury 1228
 - 53.2.3 Robotic Therapy for the Upper Extremity 1229
 - 53.2.4 Robotic Therapy for Walking 1231
- 53.3 Aids for People with Disabilities 1235
 - 53.3.1 Grand Challenges and Enabling Technologies 1235
 - 53.3.2 Types and Examples of Assistive Rehabilitation Robots 1236
- 53.4 Smart Prostheses and Orthoses 1240
 - 53.4.1 Grand Challenges and Roadblocks 1240
 - 53.4.2 Targeted Reinnervation 1240
 - 53.4.3 Brain–Machine Interfaces 1241
 - 53.4.4 Advances in Neural Stimulation 1241
 - 53.4.5 Embedded Intelligence 1242
- 53.5 Augmentation for Diagnosis and Monitoring 1242
 - 53.5.1 Introduction: Grand Challenges and Enabling Technologies 1242
 - 53.5.2 Smart Clinics with Automated Health Care Monitoring and Care 1243
 - 53.5.3 Home-Based Rehabilitation Monitoring Systems 1243
 - 53.5.4 Wearable Monitoring Devices 1244
- 53.6 Safety, Ethics, Access, and Economics 1244
- 53.7 Conclusions and Further Readings 1245
- References** 1246

54 Domestic Robotics

<i>Erwin Prassler, Kazuhiro Kosuge</i>	1253
54.1 Cleaning Robots	1254
54.1.1 The Task and Its Context	1254
54.1.2 Technical Challenges	1255
54.1.3 Domestic Floor-Cleaning Robots	1258
54.1.4 Pool-Cleaning Robots	1266
54.1.5 Window-Cleaning Robots	1267
54.1.6 Old Problems, New Technologies	1269
54.2 Lawn-Mowing Robots	1271
54.3 Smart Appliances	1273
54.3.1 Ironing Robots	1273
54.3.2 Intelligent Refrigerators	1274
54.3.3 Digital Wardrobes	1274
54.4 Smart Homes	1275
54.5 Domestic Robotics: It Is the Business Case Which Matters	1279
54.6 Conclusions and Further Reading	1280
References	1280

55 Robots for Education

<i>David P. Miller, Illah R. Nourbakhsh, Roland Siegwart</i>	1283
55.1 The Role of Robots in Education	1284
55.1.1 Interest in Science and Engineering	1284
55.1.2 Teamwork	1284
55.1.3 Problem Solving	1284
55.2 Educational Robot Tournaments	1285
55.2.1 Origins	1285
55.2.2 A Taxonomy of Robot Tournaments	1285
55.2.3 The Entertainment Link	1286
55.2.4 Tournaments for Education	1287
55.3 Education Robot Platforms	1287
55.4 Education Robot Controllers and Programming Environments	1290
55.4.1 Robot Controllers	1290
55.4.2 Edutainment Programming Environments	1292
55.5 Robots and Informal Learning Venues (Museums)	1292
55.5.1 Tour-Guide Robot Examples	1292
55.5.2 Lessons Learned	1295
55.6 Educational Evaluation of Robot Programs	1296
55.6.1 Design-Time Assessment	1296
55.6.2 Formative and Summative Evaluation	1296
55.7 Conclusions and Further Reading	1298
References	1298

Part G Human-Centered and Life-Like Robotics

56 **Humanoids**
Charles C. Kemp, Paul Fitzpatrick, Hirohisa Hirukawa, Kazuhito Yokoi, Kensuke Harada, Yoshio Matsumoto 1307

56.1 Why Humanoids? 1307

56.1.1 The Human Example 1308

56.1.2 The Pleasing Mirror 1308

56.1.3 Understanding Intelligence 1308

56.1.4 Interfacing with the Human World 1308

56.1.5 Interfacing with People 1309

56.1.6 Entertainment, Culture, and Surrogates 1310

56.2 History and Overview 1310

56.2.1 Different Forms 1311

56.2.2 Different Degrees of Freedom 1311

56.2.3 Different Sensors 1311

56.2.4 Other Dimensions of Variation 1312

56.3 Locomotion 1312

56.3.1 Bipedal Locomotion 1312

56.3.2 Falling Down 1313

56.3.3 Sensing for Balance 1314

56.3.4 Localization and Obstacle Detection 1314

56.4 Manipulation 1315

56.4.1 The Arm and Hand 1315

56.4.2 Sensing for Manipulation 1316

56.4.3 Rhythmic Manipulation 1317

56.4.4 Cooperative Manipulation 1317

56.4.5 Learning and Development 1318

56.5 Whole-Body Activities 1318

56.5.1 Coarse Whole-Body Motion 1319

56.5.2 Generating Dynamically Stable Motions 1321

56.5.3 Generating Whole-Body Motions from Operational Point Motions 1322

56.5.4 Generating Motions when in Contact with an Object 1324

56.6 Communication 1325

56.6.1 Expressive Morphology and Behavior 1325

56.6.2 Interpreting Human Expression 1327

56.6.3 Alternative Models for Human–Robot Communication 1329

56.7 Conclusions and Further Reading 1329

References 1329

57 **Safety for Physical Human–Robot Interaction**
Antonio Bicchi, Michael A. Peshkin, J. Edward Colgate 1335

57.1 Motivations for Safe pHRI 1336

57.2 Safety for Hands-Off pHRI 1337

57.3 Design of Intrinsically Safe Robots 1338

57.4 Safety for Hands-On pHRI 1341

57.5	Safety Standards for pHRI	1345
57.6	Conclusions	1346
	References	1346

58 Social Robots that Interact with People

	<i>Cynthia Breazeal, Atsuo Takanishi, Tetsunori Kobayashi</i>	1349
58.1	Social Robot Embodiment	1350
58.2	Multimodal Communication	1352
58.2.1	Robots that Express Paralinguistic Information	1352
58.2.2	Robots that Understand Paralinguistic Information	1353
58.2.3	Group Conversation	1355
58.2.4	Communication in Collaboration	1355
58.3	Expressive Emotion-Based Interaction	1356
58.3.1	Kismet: Inspiration from Developmental Psychology	1356
58.3.2	WE-4RII: A Model of Emotion Inspired by Motion	1357
58.4	Socio-cognitive Skills	1360
58.4.1	Shared Attention	1360
58.4.2	Emotional Empathy	1362
58.4.3	Mental Perspective Taking	1362
58.4.4	Perspective Taking in Collaboration	1363
58.5	Conclusion and Further Reading	1365
	References	1366

59 Robot Programming by Demonstration

	<i>Aude Billard, Sylvain Calinon, Rüdiger Dillmann, Stefan Schaal</i>	1371
59.1	History	1372
59.2	Engineering-Oriented Approaches	1374
59.2.1	Learning a Skill	1374
59.2.2	Incremental Teaching Methods	1382
59.2.3	Human-Robot Interaction in PbD	1383
59.2.4	Joint Use of Robot PbD with Other Learning Techniques ...	1385
59.3	Biologically-Oriented Learning Approaches	1386
59.3.1	Conceptual Models of Imitation Learning	1386
59.3.2	Neural Models of Imitation Learning	1386
59.4	Conclusions and Open Issues in Robot PbD	1389
	References	1389

60 Biologically Inspired Robots

	<i>Jean-Arcady Meyer, Agnès Guillot</i>	1395
60.1	General Background	1395
60.2	Bio-inspired Morphologies	1396
60.3	Bio-inspired Sensors	1398
60.3.1	Vision	1398
60.3.2	Audition	1399
60.3.3	Touch	1400

- 60.3.4 Smell 1402
 - 60.3.5 Taste 1402
 - 60.3.6 Internal Sensors 1402
- 60.4 Bio-inspired Actuators 1402
 - 60.4.1 Locomotion 1402
 - 60.4.2 Grasping 1407
 - 60.4.3 Drilling 1408
- 60.5 Bio-inspired Control Architectures 1408
 - 60.5.1 Behavior-Based Robotics 1408
 - 60.5.2 Learning Robots 1409
 - 60.5.3 Evolving Robots 1410
 - 60.5.4 Developing Robots 1411
- 60.6 Energetic Autonomy 1412
- 60.7 Collective Robotics 1413
- 60.8 Biohybrid Robots 1415
- 60.9 Discussion 1417
- 60.10 Conclusion 1418
- References** 1418

61 **Evolutionary Robotics**

- Dario Floreano, Phil Husbands, Stefano Nolfi*..... 1423
- 61.1 Method 1423
- 61.2 First Steps 1424
 - 61.2.1 Evolution of Neural Controllers for Walking 1426
- 61.3 Simulation and Reality 1428
- 61.4 Simple Controllers, Complex Behaviors 1429
- 61.5 Seeing the Light 1431
 - 61.5.1 Coevolution of Active Vision and Feature Selection 1432
- 61.6 Computational Neuroethology 1434
 - 61.6.1 Emergence of Place Cells 1435
 - 61.6.2 Spiking Neurons 1436
 - 61.6.3 GasNets 1438
- 61.7 Evolution and Learning 1439
 - 61.7.1 Learning to Adapt to Fast Environmental Variations 1440
 - 61.7.2 Evolution of Learning 1441
- 61.8 Competition and Cooperation 1442
 - 61.8.1 Coevolving Predator and Prey Robots 1442
 - 61.8.2 Evolving Cooperative Behavior 1443
- 61.9 Evolutionary Hardware 1444
 - 61.9.1 Evolvable Hardware Robot Controllers 1444
 - 61.9.2 Evolving Bodies 1445
- 61.10 Closing Remarks 1447
- References** 1447

62 Neurorobotics: From Vision to Action

<i>Michael A. Arbib, Giorgio Metta, Patrick van der Smagt</i>	1453
62.1 Definitions	1453
62.2 Neuroethological Inspiration	1454
62.2.1 Optic Flow in Bees and Robots	1455
62.2.2 Visually Guided Behavior in Frogs and Robots	1456
62.2.3 Navigation in Rat and Robot	1457
62.2.4 Schemas and Coordinated Control Programs	1459
62.2.5 Saliency and Visual Attention	1461
62.3 The Role of the Cerebellum	1462
62.3.1 The Human Control Loop	1462
62.3.2 Models of Cerebellar Control	1463
62.3.3 Cerebellar Models and Robotics	1466
62.4 The Role of Mirror Systems	1467
62.4.1 Mirror Neurons and the Recognition of Hand Movements ..	1467
62.4.2 A Bayesian View of the Mirror System	1470
62.4.3 Mirror Neurons and Imitation	1473
62.5 Extrodution	1474
62.6 Further Reading	1475
References	1475

63 Perceptual Robotics

<i>Heinrich H. Bülthoff, Christian Wallraven, Martin A. Giese</i>	1481
63.1 Overview	1481
63.2 Example-Based Object Representations	1483
63.2.1 Perceptual and Computational Basis of Object Representations	1483
63.2.2 Neural Representations in Object Recognition	1485
63.2.3 Object Recognition: Lessons from Computer Vision	1486
63.2.4 Object Learning and Recognition for Perceptual Robotics ..	1488
63.3 Example-Based Movement Representations	1490
63.3.1 Recognition of Complex Movements in Visual Cortex	1490
63.3.2 Example-Based Motion Recognition in Robots and Computer Vision	1491
63.4 Example-Based Synthesis Models: From Faces to Movements	1492
63.4.1 Face Synthesis: Realistic Faces for Avatars and Robots	1492
63.4.2 Example-Based Trajectory Synthesis	1493
63.5 Conclusions and Further Reading	1494
References	1495

64 Roboethics: Social and Ethical Implications of Robotics

<i>Gianmarco Veruggio, Fiorella Operto</i>	1499
64.1 A Methodological Note	1501
64.2 Specificity of Robotics	1502

64.3	What Is a Robot?	1502
64.3.1	Robots Are Nothing Else But Machines	1502
64.3.2	Robots (and Technology in General) Have an Ethical Dimension	1502
64.3.3	Robots as Artificial Moral Agents (AMA)	1503
64.3.4	Robots: the Evolution of a New Species	1503
64.4	Cultural Differences in Robot's Acceptance	1503
64.5	From Literature to Today's Debate	1503
64.6	Roboethics	1504
64.7	Ethics and Morality	1505
64.8	Moral Theories	1505
64.9	Ethics in Science and Technology	1506
64.10	Conditions for Implementation	1507
64.11	Operativeness of the Principles	1507
64.12	Ethical Issues in an ICT Society	1507
64.13	Harmonization of Principles	1509
64.14	Ethics and Professional Responsibility	1510
64.15	Roboethics Taxonomy	1511
64.15.1	Humanoids	1511
64.15.2	Artificial Body	1511
64.15.3	Industrial Robotics	1513
64.15.4	Adaptive Robot Servants	1513
64.15.5	Distributed Robotic Systems	1514
64.15.6	Outdoor Robotics	1514
64.15.7	Surgical Robotics	1515
64.15.8	Biorobotics	1515
64.15.9	Biomechatronics	1515
64.15.10	Health Care and Quality of Life	1515
64.15.11	Military Robotics	1516
64.15.12	Educational Robot Kits	1517
64.15.13	Robot Toys	1518
64.15.14	Entertainment Robotics	1518
64.15.15	Robotic Art	1518
64.16	Conclusions and Further Reading	1519
	References	1522
	Acknowledgements	1525
	About the Authors	1527
	Detailed Contents	1555
	Subject Index	1591

Subject Index

3-D sensing (at low cost) 1269
 3-D sensor 973
 3T 191

A

AAAI Robot Challenge 200
 absolute
 – coordinate frame 707
 – localization (at low cost) 1269
 – orientation 707
 – position 707
 – positioning 1255
 abstraction layers 845
 acceleration radius 238, 239
 acceleration-resolved control 167, 177
 accelerometers 732
 accessibility 1510
 accuracy 1510
 – analysis 273
 acoustic baseline 996, 1000
 Acrobot surgical robot 1206, 1207, 1210
 action
 – joint 1355
 action recognition
 – passive 926
 action selection 895, 902–904
 activation
 – behavior-based 895, 896, 898–901, 903, 904
 active
 – compliance 165
 – compliant motion 631
 – cord mechanism (ACM) 262
 – fine motion planning 631
 – perception 952
 – sensing 634
 – sensor network (ASN) 602
 – stiffness 165–167
 – vision 1432–1434
 activities of daily living (ADLs) 1223, 1225
 actuated grippers 720
 actuation 728
 actuator 732
 – effort sensing 459
 adaptation 1395, 1417

adaptive
 – behavior 1439, 1447
 – control 147, 306
 – cruise control (ACC) 1187
 – seek control logic (ASCL) 1266
 admittance 724
 – control 163, 169, 170
 – matrix 630
 advanced
 – driver assistance system (ADAS) 1177, 1189
 – highway systems (AHS) 1191
 – rings 513
 – servomanipulator (ASM) 744, 1112
 – technology for large structural systems (ATLSS) 1094
 aerial robotics 1009, 1010
 Aesop surgical robot 1204
 affective
 – dimensions 1356
 – prosody 1356
 – state 1356
 Afghanistan Mine Action Center (MACA) 1106
 agent
 – remote 191
 agriculture 1066
 Aibo 1427
 airborne collision avoidance systems (ACAS) 1022
 Alexander the Great 989
 algorithmic singularities 258
 ALIVE 989, 1002
 ALLIANCE 924, 929
 American National Standards Institute (ANSI) 1345
 analytic Jacobian 248
 android robot 1351
 angle random walk 482
 angle-axis 12
 antenna 458
 anterior inferotemporal cortex (AIT) 1486
 anterior interparietal area (AIP) 1495
 antipersonnel (AP) 1104
 antitank mines (AT) 1104
 antivehicle (AV) 1104
 anytime algorithm 214
 appeal 1506

appliances
 – smart 1273
 application specific integrated circuit (ASIC) 429
 applications of CTFM 510
 approaches
 – bottom-up 1418
 – holistic 1417
 – integrative 1396
 – reductionist 1417
 – top-down 1417
 appropriate design 280
 arbitration 190
 architecture 189, 892, 894, 922
 – behavior-based 902, 903
 – three-layer 893, 894
 area coverage 1256
 ARIES 1000
 Aristotle 989
 ARM 1237
 ARM-In 1231
 armoured (or articulated) face conveyor (AFC) 1141
 articulated-body algorithm (ABA) 53
 artifacts 500
 artificial
 – constraint 174
 – evolution 1437, 1444
 – intelligence (AI) 207
 – moral agents (AMA) 1503
 – neural network (ANN) 1373
 Asimov's three laws of robotics 1503
 ask-backs 1354
 assembly line 966
 assembly motion
 – compliant motion 628
 assistive robot service manipulator 1237
 assistive technology 722
 Association for the Advancement of Artificial Intelligence (AAAI) 1286
 assumed modes 311
 ASyMTRe 929
 ATLANTIS 191
 atomic force microscope (AFM) 418

attention
 – joint 1353
 – shared 1355, 1361
 attentional focus 1361
 attention-monitoring behavior 1361
 attractor
 – behavioral 1431
 audition 1399
 augmented Jacobian 257
 augmented reality (AR) 1374
 autism 1224, 1285
 – spectrum disorder (ASD) 1285
 automated
 – bus rapid transit (ABRT) 1193
 – drilling 1137
 – guided vehicle (AGV) 968
 – highway 1193
 – vehicles 1191
 automatic steering 1070
 automotive 722
 autonomous mental development (AMD) 1311
 Autonomous Ocean Sampling Network (AOSN) 1003
 autonomous robot architecture (AuRA) 190
 autonomous space transport robotic operations (ASTRO) 1037
 autonomous underwater vehicle (AUV) 988, 1516
 autonomous vehicle 1109–1121
 autonomy 1395, 1417
 – behavioral 1412
 – shared 748, 1035
 auxiliary surgical support robots 1204

B

back hoe 1144
 back-channel responses 1353
 back-drivability 720, 724
 backstepping control 306
 backward-difference equation 154
 balancer 278, 974
 bang-bang-control 120
 basic emotions 1359
 BASIC stamp 1291
 basis behaviors
 – behavior-based 897
 basis facial postures 1356
 basket cell 1464
 batch estimation 94
 battlebots 1286
 Baumgarte stabilization 59

Bayes
 – rule 586
 Bayes'
 – network 211
 Bayesian networks (DBNs)
 – dynamic 212
 beam
 – bending 308
 – torsion 308
 beam pattern
 – sonar 494
 bearing estimation 506
 bee 1455
 behavior 190
 – adaptive 1439, 1447
 behavioral
 – attractor 1431
 – autonomy 1412
 – control 196
 – homeostasis 1356
 behavior-based 896, 899
 – activation 895–904
 – basis behaviors 897
 – coordination 895
 – fuzzy systems 902, 905
 – history of behavior use 900
 – interaction dynamics 893, 895, 897–900
 – learning 898, 902
 – misconceptions 896
 – multi-robot systems 897–899, 902
 – network 896–899
 – planning 896, 898, 899, 904
 – principles 894, 901
 – reasoning 896, 903
 – reinforcement learning 899, 901
 – representations 896, 897, 903
 – robotics 190, 383
 – systems 891, 893, 896
 – vision systems 903
 Beidou 485
 Bellman equation 221
 beneficence 1506
 Berkeley lower-extremity exoskeleton (BLEEX) 781
 BEST 1285
 best linear unbiased estimator (BLUE) 340
 bias repeatability 482
 bilateral 742
 Bill McNeely 723
 Bi-Manu-Track 1231
 bimorph actuator 734
 bin-picking 973
 bioethics 1508

bio-inspired
 – actuators 1396
 – control 1396
 – morphologies 1396
 – sensors 1396
 biomimetic 515
 – robotics (BIOROB, HUMANOIDS) 1373
 BION 1242
 bipedal robot 1427
 blind-spot detection 1186
 block cave mining 1133
 body-affect mappings 1362
 body-weight supported treadmill training (BWSTT) 1232
 Boeing 723
 Botball robot tournament 1286
 bounding spheres 729
 box pushing 934
 bracing strategy 308
 broad-acre 1071
 broadcast 194
 – of local eligibility (BLE) 902, 931
 Brockett's theorem 815
 browser 759
 Brunovsky canonical form 142
 bulldozer 1144
 bulldozing resistance 1056
 buoyancy 993
 – center of 993
 bus rapid transit (BRT) 1193
 – automated (ABRT) 1193
 business case 1279

C

cable-driven robot 288
 cables 726
 calculation
 – force 728
 capstan 726
 capture and berthing 1035
 car assembly 971
 car body assembly 970
 Care-O-bot 1238
 carrier sense multiple access (CSMA) 761
 cartesian robot 968
 cartesian-space stiffness matrix 240
 caster wheel 394, 396
 cellular automata 912
 cellular robot (CEBOT) 915
 center compliance
 – remote (RCC) 629

- center of
 - buoyancy 993
 - compliance 165
 - gravity (CoG) 991, 1322
 - rotation (CoR) 651, 662
 - stiffness 165
- center of compliance (RCC)
 - remote 166
- center of motion (RCM)
 - remote 1204, 1212
- center pivot 1071
- central pattern generator (CPG) 368, 1317
- centroid
 - contact 733
- cerebellar model articulation controller (CMAC) 1462
- cerebellum 1453, 1462
- Chai3D 724
- chain topology
 - kinematic 230
- chained-form system 807, 833
- chaotic behavior 365
- characteristic length 237
- characteristic output vector (COV) 601
- charge-coupled device (CCD) 418, 428
- charter 1507
- Charles' Theorem 14
- chemotaxis 1402
- circuit switching 762
- citrus 1072
- cleaning robots 1254, 1258
- C-Leg 1242
- client–server 188, 194
- climbing fiber (CF) 1464
- clinical gait analysis (CGA) 786
- closed chains 269
- closed kinematic chains 121
- closed kinematic loop 57, 327
- closed loop execution and recovery (CLEaR) 193
- closed-loop inverse kinematics (CLIK) 255
- closest point (CP) 534
- cluster bombs (CB) 1104
- clustering 100
- coal mining 1127, 1140
- coarse-acquisition 485
- Cobots (collaborative robots) 1343
- code 1506
- coefficient of restitution 1051
- cognition 891
- cognitive-affective architecture 1356
- cognizant failure 196
- collaboration 1363
- Collaborative Observatory for Nature Environments (CONE) 769
- collaborative operation 975
- collective 926
 - robotics 1413
- collision 728
 - avoidance 1185
 - detection 727, 729
 - mitigation 1185
 - warning 1184
- command shaping 313
- commercial off-the-shelf (COTS) 599
- Commission de Energie Atomique (CEA) 1112
- common gateway interface (CGI) 763
- common ground 1355
- common object request broker architecture (CORBA) 194, 769
- communication 925, 943, 1012, 1224
 - for control 950
 - for perception 951
- comparison of CTFM and pulse-echo sonar 509
- compass 995
- competition and cooperation 1442
- complementarity condition 366
- complementarity problem (CP) 656
- compliance 975
 - active 165
 - center of 165
 - control 163, 170, 300
 - frame 163, 174
 - matrix 165, 174
- compliant
 - contact 648
 - environment 173, 179, 180
 - motion execution 634
 - motion planning 632
- compliant motion
 - active 631
- component modeling 995
- composite-rigid-body algorithm (CRBA) 54
- compression 308
- computational fluid dynamics 995
- computational neuroethology 1434
- computed
 - tomography (CT) 1211, 1212
 - torque 303
 - torque approach 368
 - torque control 143
- torque method 802
- ZMP 373
- computer
 - aided design (CAD) 722
 - ethics (CE) 1508
 - numerically controlled (CNC) 1083
 - torque control 401
- computer aided design (CAD) 723, 979, 1082
- models 729
- computer professional for social responsibility (CPSR) 1508
- computer simulations 1165
- computer-aided remote driving (CARD) 1040
- computer-aided tomography (CAT) 1515
- computer-integrated surgery (CIS) 1200
- concurrent mapping and localization (CLM) 1001
- condition number 250, 336
- conditioning
 - kinematic 724
- configuration space 110
 - manipulation 622
 - obstacles 625
 - topology 625, 626
- consequentialism 1505
- conservative congruence transformation (CCT) 277
- constrained modes, of joint 19
- constrained motion 162
- constraint
 - artificial 174
 - Jacobian 171, 172
 - kinematic 162, 171
 - nonholonomic 828
 - (selection) matrix 654
- constraint-task Jacobian 258
- construction
 - automation 1079, 1080
 - elementary processes 1086
 - industry 1079
 - phases of a project 1081
 - programmable machine 1082
 - project 1080
 - robots 1080
 - stakeholders 1081
- construction robotics 1081
 - barriers to 1086
 - categories of 1082
- contact 732
 - centroid 733
 - compliant 648

- display 733
- dynamics 1050
- formation (CF) 633
- interface 647, 659
- kinematics 648
- location sensing 461
- mode 650
- model 647, 674
- state identification 633
- stiffness 665
- vibration 732
- virtual manipulator 176
- contact state
 - principal contact (PC) 629
- contextual inquiry 1296
- continuation 272
- continuous activity scheduling,
 - planning, execution and replanning (CASPER) 193
- continuous time 731
- continuous-transmission frequency
 - modulation (CTFM) 508
- continuum Jacobian 264
- contract net protocol (CNP) 931
- contraction mapping 300
- contractualism 1505
- control
 - adaptive 147, 306
 - admittance 163, 169, 170
 - basis 621
 - behavioral 196
 - command interpreter (CCI) 601
 - damping 163, 170
 - decentralized 137
 - feedback 799
 - for perception 951
 - force 724
 - hybrid 369
 - motion 134, 979
 - multiobjective 620
 - null-space projection 620
 - of nonholonomic systems 806, 818
 - operational space 618
 - position and force 619
 - redundant manipulator 619
 - remote 760
 - shared 742, 746, 748
 - system 118
 - task-level 618
- controllability 403
- controllable forces and velocities 677, 679
- controlled traffic 1070
- convention 1507

- conversational 1352
 - roles 1355
- cooperation 943
 - cooperative manipulation planning 639
- cooperative 922
 - manipulation 914, 934
 - manipulator 701
 - multi-arm systems 711
 - task space 707
- cooperative intelligent real-time
 - control architecture (CIRCA) 193
- cooperative multirobot observation of
 - multiple moving target (CMOMMT) 934
- coordinate frame
 - absolute 707
- coordination 943
 - behavior-based 895
 - decentralized 946
- CORAL 998
- cord mechanism (ACM)
 - active 262
- Coriolis acceleration 481
- Coriolis and centrifugal forces 134
- CosmoBot 1239
- Coulomb 729
 - friction 652
- Coulomb friction model 685
- counterweights 278
- coupled layered architecture for robot
 - autonomy (CLARAty) 192
- coupling
 - stiffness 165
 - virtual 728, 729, 731
- coverage 933
 - configuration protocol (CCP) 954
- Cramér–Rao lower bound (CRLB) 951
- crawl gait 379
- crawling 1402
- credit assignment problem 1465
- creeping gait 379
- criteria of Nyquist 730
- cruise control (ACC)
 - adaptive 1187
- C-space 110
- CTFM (continuous-transmission
 - frequency modulation) 491
 - range discrimination and resolution 509
 - sonar 508
 - TOF estimation 508
 - transmission coding 508
- current amplifiers 726
- curvature sensing 469

- curvature velocity method 839
- curvilinear abscissa 803
- cutaneous 719
- cutaneous (tactile) 720
- cyclicality 260
- cylindrical algebraic decomposition 126
- cylindrical joint 21

D

- damped least squares 337
- damping 726
 - control 163, 170
 - factor 253
 - ratio 169
- data association 100
- data communication 1120
- data fusion (DDF)
 - decentralized 604
- datcom 992
- daVinci 1010
 - surgical robot 1202–1205, 1213–1215
- DC motor 137
- deactivation and decommissioning (D&D) 1114
- decentralized
 - control 137
 - coordination 946
 - data fusion (DDF) 604
- declaration 1506
- deconvolution 466
- decoupled architecture 231
- decoupling control 302
- dedicated short-range
 - communications (DSRC) 1180, 1194
- deep reactive ion etching (DRIE) 431
- defective grasp 679
- Defense Sciences Office (DSO) 1240
- degree of
 - mobility 397
 - steerability 397
- degrees of freedom (DOF) 20, 433, 1048, 1311
- deictic gestures 1354
- deliberative systems 892, 894, 899
- delta robot 231, 270, 976
- demining 1103, 1106–1109, 1114, 1121
- demographics 1224
- Dempster–Shafer theory 597

- Denavit–Hartenberg (DH) 324
 - convention 23
 - parameters 230, 324
 - Denning ring 512
 - deontology 1505
 - Department of Defense (DoD) 762
 - depth sensor 995
 - design 279, 722
 - considerations 680
 - for assembly 967
 - philosophy 230
 - designing a robotic architecture 202
 - desirable properties 679
 - desktop vocational assistant robot (DeVAR) 1225
 - detection
 - vehicle 1183
 - development 1396
 - processes 1411
 - dexterity
 - global 238
 - indices 235
 - local 235
 - difference of Gaussians (DOG) 553
 - differential flatness 405, 834
 - digital
 - factory 983
 - maps 1180
 - wardrobe 1274
 - dilution of precision (DOP) 487
 - dimensionless space 237
 - dimensions
 - affective 1356
 - dip-pen nanolithography (DPN) 422
 - direct control 742, 746
 - directivity pattern 494
 - directors of the laboratories (JDL)
 - joint 598
 - disaster response 1012
 - discontinuous contact 730
 - discrete Fourier transform (DFT) 509
 - display
 - contact 733
 - tactile 731
 - vibration 722
 - distance from singularities 249
 - distributed 921
 - autonomous robotics system (DARS) 918
 - field robot architecture (DFRA) 600
 - macro-mini actuation (DM2) 1340
 - robot architecture (DIRA) 923, 925
 - disturbances
 - environmental 991
 - dogs 1105–1107
 - Doppler velocity log (DVL) 996
 - drag force 993
 - dragline 1144
 - drawbar pull 1055
 - drift 482
 - drilling 1408
 - automated 1137
 - driver
 - assistance 1184
 - fatigue 1189
 - impairment 1189
 - inattention 1189
 - monitoring 1188
 - protection 1190
 - workload 1190
 - dual-arm robot 968
 - duality
 - kinematic 659
 - duty factor 379
 - dynamic
 - Bayesian networks (DBNs) 212
 - equations 677
 - feedback linearization 304, 405
 - manipulability 725
 - matrix 241
 - matrix invariance 241
 - multisensor system control 603
 - singularity 1047
 - sinkage 1054
 - state machine (DSM) 1445
 - tactile sensing 460, 470
 - task workspace (DTW) 903
 - window approach (DWA) 842
 - dynamic walker
 - passive 1427
 - dynamical equilibrium 367
 - dynamically-consistent inverse of the Jacobian matrix 265
 - dynamics 279
 - contact 1050
 - inverse 50, 167, 178, 179, 293, 313
 - dynamics and equilibrium 676
 - dynamics control
 - inverse 141
- ## E
- echo waveform
 - coding 503
 - processing 506
 - economic situation 1254
 - eddy current metal detector 1106, 1113
 - education 722
 - educational
 - evaluation 1296
 - robotics 1284
 - effects
 - motion 513
 - ego-motion 91
 - eigenvalue decomposition 235
 - EKF SLAM 875
 - elastic
 - bands 637, 847
 - roadmap 637
 - strips 637
 - elasticity
 - joint 289
 - elastostatic performance 239, 241
 - electrical master–slave manipulators (EMSs) 1111
 - electrical stimulation (ES) 1231
 - electroencephalogram (EEG) 1241
 - electromyography (EMG) 1231
 - electron microscope (EM) 417
 - electron-beam-induced deposition (EBID) 435
 - electronic distance measuring (EDM) 1083
 - electronics controller unit (ECU) 1036
 - electrostatic 498
 - elementary operator (EO) 1373
 - elevation model 537
 - embedded computers 944
 - emergence 895, 897, 898
 - emergency stop 1346
 - emotional
 - appraisal 1357
 - coefficient matrixes 1357
 - empathy 1362
 - emotion-based interaction 1350, 1356
 - EMS-vision system 603
 - encode 727
 - encoder 725
 - end-effector 980
 - motion 136
 - endolite adaptive prosthesis 1242
 - endoluminal
 - robots 1214
 - surgery 1213
 - energetic autonomy 1396, 1412
 - energy leaks 731
 - energy stability margin 385
 - entertainment 722

envelope displays 1353
 environment
 – compliant 173, 179, 180
 – virtual 720, 727, 730
 environmental
 – disturbances 991
 – model 89
 – monitoring 955
 equation
 – of emotion 1357
 – of motion 134
 equilibrium
 – dynamical 367
 error recovery 1256
 estimation 87
 – linear models 94
 – process 93
 ethernet 761, 969
 ethics 1505
 ethnography 1297
 Euler angles 11, 990
 Euro Geostationary Navigation
 Overlay Service (EGNOS)
 487
 European robotic arm (ERA) 1034
 European Space Agency (ESA)
 1034
 evaluation 1165
 event detection sensing 470
 evidential reasoning 597
 evolution 1396
 – artificial 1437, 1444
 – Hebbian learning 1434, 1440,
 1441
 – learning rule 1429, 1441–1443
 – of learning 1429, 1439–1441
 evolutionary
 – algorithm 1410, 1423, 1437,
 1445, 1446
 – robotics 1423–1447
 evolvable hardware 1444–1446
 – field programmable gate array
 (FPGA) 1445
 – programmable logic device (PLD)
 1445
 examples 687
 excavator 1113, 1144
 execution support language (ESL)
 198
 executive 197
 exoskeleton 720, 1416
 expectation maximization (EM)
 101, 220, 1472
 exploratory procedure (EP) 722
 explosive
 – hazard 1142

 – ordnance disposal (EOD) 1101,
 1103
 – placement 1137, 1140, 1147
 – vapor detector 1107
 exponential stabilization 303
 expressive vocalization system
 1356
 extended
 – Jacobian 257
 – Kalman filter (EKF) 96, 591, 859,
 871
 – physiological proprioception (EPP)
 1240
 external force 704
 – manipulability ellipsoid 710
 external velocity manipulability
 ellipsoid 710
 exteroception 87
 extrastriate body part area (EBA)
 1490
 extravehicular activity (EVA) 746,
 1032

F

face space 1492
 face-to-face interactions 1362
 facial expression 1356, 1359
 factor
 – damping 253
 factory
 – digital 983
 false-belief tasks 1363
 FARS model 1468
 fast Fourier transform (FFT) 311,
 509
 feasible
 – solution of wrench (FSW) 378
 – velocities 249
 feature
 – extraction 88
 – frame 176
 – histograms 973
 – selection 1432, 1433
 feedback 760
 – control 799
 – dynamic linearization 304, 405
 – haptic 720, 1215
 – hybrid 818
 – linearizability 834
 – linearization 142, 301, 404
 – motion planning 635
 – of motor variables 298
 – planning 634
 – vibration 731, 732
 feedforward motion 181
 feedforward/feedback control 304
 fiber to the home 763
 field programmable gate array
 (FPGA) 1445
 fieldbus 969
 field-emission SEM (FESEM) 433
 figure of merit 725
 filtering 587
 fine motion planning
 – active 631
 finger 735
 fingerpad 733
 finite element (FE) 729
 finite-state
 – acceptor (FSA) 190
 – machine (FSM) 1295
 – transducer (FST) 1353
 firewall 762
 FIRST 1285
 first-in first-out (FIFO) 762
 first-order predicate logic (FOPL)
 208
 fixed angles 12
 fixed-point stabilization 801
 fixture
 – and contact 664
 – virtual 728, 1201, 1206–1208
 flatness 295
 flexibility 980
 flexible manufacturing system (FMS)
 968
 FlexPicker 231
 floating base 22
 flocking 933
 floor
 – smart 1269
 flying 1406
 foot rotating indicator (FRI) 378
 foraging 933
 force
 – calculation 728
 – control 724
 – reflection 1111
 – sensors 726
 – shading 729
 force closure 684, 685
 – tests 686
 force sensing resistor (FSR) 460
 force/motion control
 – hybrid 163, 177
 force-position control 178
 force-torque sensing 459
 force-torque sensor 163, 974
 force-velocity control 178
 Forestix 1075
 forestry 1066

fork lift 968
 form closure 650, 681
 – requirements 683
 – test in the plane 684
 – tests 683
 formation 933
 – contact (CF) 633
 formative evaluation 1296
 forward
 – dynamics 53, 54
 – instantaneous kinematics 29
 – kinematics 26, 272, 728
 forwarder 1066
 four wheel drive (4WD) 1133
 free modes of joint 19
 free space voxels 532
 free vibration 241
 free-floating manipulator 1045
 French Atomic Energy Commission (CEA) 1225
 Frénet frame 803
 friction 171, 724
 – cone 652
 – limit surface 661
 – models 729
 friction models 685
 frictional
 – ambiguity 655
 – form closure 685
 – inconsistency 655
 frog 1456
 – muscles 1415
 front-end loaders 1144
 Froude number 386
 fuel cells 1271
 full-state feedback 300
 functional
 – magnetic resonance imaging (fMRI) 1491
 – neural stimulation (FNS) 1224
 Fututech 1073
 fuzzy logic 597, 1373
 fuzzy systems
 – behavior-based 902, 905

G

gait
 – diagram 379
 – generation 912
 – training 1232
 Galileo positioning system 484
 gantry 976
 gastrointestinal (GI) endoscopy 1213
 Gaussian mixture
 – model (GMM) 1378, 1380
 – regression (GMR) 1378, 1380, 1383
 Gaussian processes (GP) 1378
 Gauss–Markov estimate 340
 Gauss–Newton nonlinear estimation 322
 gaze direction 1353
 generalized
 – Jacobian matrix (GJM) 1046
 – least-squares estimate 340
 generalized-inertia ellipsoid 237
 genetic algorithm 1424, 1426
 – artificial genome 1424
 – cellular encoding 1427
 – chromosome 1424, 1432
 – fitness function 1424, 1425, 1435, 1437, 1446
 – genotype 1426, 1427, 1440
 – mutation 1424, 1428, 1437, 1446
 – phenotype 1440
 genetic programming 1446
 Geneva International Center for Humanitarian Demining (GICHD) 1107
 GENTLE 1231
 geology-based guidance 1139
 geometric Jacobian 248
 geon structural description (GSD) 1483
 George Devol 964
 geostationary Earth orbit (GEO) 1035
 German Aerospace Agency (DLR) 1034
 global
 – asymptotic stability (GAS) 140
 – conditioning index 238
 – navigation satellite system (GNSS) 1015, 1178
 – navigation satellite system landing system (GLS) 487
 – navigation satellite systems (GNSS) 1178
 – navigation systems (GNS) 1020
 – positioning system (GPS) 996, 1020, 1066, 1083, 1147
 globalnaya navigatsionnaya sputnikovaya sistema (GLONASS) 484
 goal-contact relaxation (GCR) 633
 goal-frame set 233
 Golem project 1446

golgi
 – cell 1464
 – tendon organs 721
 graduate robot attending conference (GRACE) 200
 grail 576
 graphical
 – model 98
 – user interface (GUI) 599, 998
 grapple fixture 1032
 grasp
 – classifications 678
 – matrix 705
 – matrix and hand Jacobian 673
 graspable grasp 678
 grasping 671, 1408
 – contact models 674
 – controllable forces and velocities 677, 679
 – Coulomb friction model 685
 – defective grasp 679
 – design considerations 680
 – desirable properties 679
 – dynamic equations 677
 – dynamics and equilibrium 676
 – examples 687
 – force closure 684, 685
 – force closure tests 686
 – form closure 681
 – form closure requirements 683
 – form closure test in the plane 684
 – form closure tests 683
 – friction models 685
 – frictional form closure 685
 – grasp classifications 678
 – grasp matrix and hand jacobian 673
 – hyperstatic grasp 679
 – indeterminate grasp 678
 – models and definitions 672
 – planar force closure test 687
 – planar simplification of contact models 676
 – quasistatic equation 677
 – redundant grasp 678
 – restraint analysis 680
 – rigid body velocity kinematics 673
 – salisbury hand 672
 – velocity kinematics 673
 Grassmann geometry 275
 gravity
 – center of (CoG) 991, 1322
 – compensation 298
 – force 134
 grid-based model 104

Gröbner bases 272
 gross domestic product (GDP) 1080
 ground 725
 – penetrating radar (GPR) 1084,
 1106, 1108, 1109, 1113
 – support 1139, 1143
 ground-based augmentation system
 (GBAS) 488
 grounded 725
 group conversation 1355
 guidance
 – inertial 1147
 guided vehicle
 – automated (AGV) 968
 gyroscope 1010
 GZMP (generalized ZMP) 1325

H

Haar measure 125
 Hamilton–Jacobi–Bellman (HJB)
 equation 151
 Hamilton–Jacobi–Isaac (HJI)
 inequality 151
 hand–eye locomotion 1040
 handling
 – remote 1101, 1102, 1112
 hands on payload 1343
 hands-off pHRI 1337
 hands-on cooperative control 1203,
 1205–1207, 1210
 hands-on pHRI 1341
 handyboard 1291
 haptic 719
 – device (HD) 720, 731, 735
 – feedback 720, 1215
 – interaction point (HIP) 726
 – interface 719, 1119
 – loop 720
 – paddle 726
 – rendering 720, 727, 735
 – temperature display 735
 – walker 1233
 hard
 – contact 727
 – finger (HF) 674
 harmonic drive 288, 967
 hazard
 – avoidance 1042
 – explosive 1142
 hazardous site 1112
 head
 – injury criterion (HIC) 1338
 – mounted display (HMD) 1119
 health and safety 1130
 HeartLander 1214

Heathkit Hero-1 1288
 helical joint 21
 helicopter 1011
 HelpMate 1238
 Hemisson 1288
 Hertzian contact 659
 heterogeneity 928
 hexapod robot 1426, 1445
 HIC (head injury criterion) 1338
 hidden Markov model (HMM) 500,
 1373, 1379, 1493
 hierarchical
 – attentive multiple models for
 execution and recognition
 (HAMMER) 1363
 – task net (HTN) 199
 – task network (HTN) 216
 higher pair 21
 high-resolution transmission electron
 microscopes (HRTEM) 433
 highwall mine 1142
 highway
 – automated 1193
 history of behavior use
 – behavior-based 900
 holonomic
 – constraints 171
 – manipulator 804
 homeostasis
 – behavioral 1356
 homes
 – smart 1275
 homogeneous 926
 – space 237
 – transformation 13
 homography 579
 Honeywell ore recovery and
 tunneling aid (HORTA) 1143
 Hooke's law 728, 729
 hopping robots 363
 horticulture 1071
 hot zone 1153
 Hubble space telescope (HST) 1057
 human
 – augmentation 974
 – control 720
 – operator (HO) 730
 – subject experiment 1355
 human-arm-like manipulators 247
 human–computer interaction (HCI)
 722, 1296
 human-in-the-loop 741
 human-machine cooperative system
 (HMCS) 1207
 humanoid 46
 humanoid robotics 1237

human–robot cooperation 974
 human–robot interaction (HRI)
 902, 903, 905, 1167, 1170, 1335,
 1349
 – cognitive (cHRI) 1335
 – physical (pHRI) 1335
 human-robot interaction (HRI)
 1384
 human-robot interaction (HRI, AISB)
 1373
 human-to-robot ratio 1170
 Hurwitz stable polynomial 812
 hybrid
 – assisted limb (HAL) 774
 – control 369
 – feedback 818
 – force/motion control 163, 177
 – position/force control 711
 – systems 893, 894, 896, 898, 902
 – systems manipulation planning
 623
 hyperbolic positioning 485
 hyperredundant manipulators 261
 – backbone 262
 – continuum structure 263
 – rigid-link structure 261
 hyperstatic grasp 679
 hypertext 759
 – markup language (HTML) 763
 – transmission protocol (HTTP)
 760

I

I3CON (industrialized, integrated,
 intelligent, construction)
 1082
 ideomotor principle 1474
 illumination scenarios 1183
 illusions
 – tactile 735
 illustrator 1352
 – cues 1353
 image-based visual servo 565
 – cylindrical coordinates 570
 – interaction matrix 565
 – stability analysis 568
 – stereo cameras 570
 imitation 1470, 1473
 Immersion Corporation 723
 immersive environment 1148
 impact 45
 – dynamics 1051
 impedance
 – control 163, 714, 1052
 – matching 1052

- importance
 - density 594
 - sampling 97
 - improvised explosive device (IED) 1104, 1107
 - incremental evolution 1442
 - independent
 - component analysis 1380
 - likelihood pool 586
 - independent-joint control 137
 - indeterminate grasp 678
 - indoor GPS (iGPS) 1083
 - indoor surfaces 537
 - inductive logic programming (ILP) 220
 - inertia matrix 134
 - inertia tensor
 - inverse 1046
 - inertial
 - guidance 1147
 - measurement unit 482
 - measurement unit (IMU) 995, 1020
 - navigation 1178
 - navigation (INS) 1143
 - parameter estimation 330
 - parameter estimation: links 332
 - parameter estimation: loads 330
 - inference
 - Bayesian 211
 - inferotemporal cortex (IT) 1486
 - informal learning 1283
 - information
 - ethics 1508
 - filter 593
 - invariants 915
 - Information Society Technologies (IST) 1194
 - informational support 1364
 - infrared detectors 1106
 - injectable stimulator 1242
 - inner-loop control 142
 - innovation 591
 - input-output-to-state stability (IOSS) 140
 - input-to-state stability (ISS) 140
 - instability 730
 - instantaneous allocation (IA) 930
 - instantaneous center of rotation (ICR) 397
 - instantaneous kinematics
 - inverse 30
 - institutionalization 1224
 - instrumental support 1364
 - instrumented LSS (ILSS) 601
 - integral control action 812
 - integrating planning and execution 199
 - integration 1130
 - intelligence (AI)
 - artificial 207
 - intelligent
 - assist device (IAD) 1337, 1341, 1345
 - assist devices (IAD) 776
 - multimode transit system (IMTS) 1194
 - refrigerator 1274
 - interaction dynamics
 - behavior-based 893–895, 897–900
 - interaction matrix 564
 - approximations 565
 - direct estimation 571
 - image-based visual servo 565
 - position-based visual servo 573
 - interaction point (HIP)
 - haptic 726
 - interactive C 1292
 - interaural
 - amplitude difference (IAD) 507
 - time difference (ITD) 507
 - interface
 - contact 659
 - haptic 719, 1119
 - interface definition language (IDL) 194, 203
 - interference rejection 511
 - intermediate haptic interaction point (IHIP) 729
 - intermediate haptic interaction points (IHIP) 728
 - internal dynamics 295
 - internal force 705
 - manipulability ellipsoid 710
 - internal friction angle 1055
 - internal velocity manipulation ellipsoid 710
 - International Campaign to Ban Landmines (ICBL) 1103
 - International Commission on Illumination (CIE) 1183
 - International Organization for Standardization (ISO) 1345
 - international space station (ISS) 1033
 - internet 759
 - communications engine (ICE) 194
 - protocol (IP) 762
 - service providers (ISP) 763
 - internet engineering task force (IETF) 1180
 - interpenetration 730
 - inter-process communication 188, 194
 - interval
 - analysis 272
 - calculus 596
 - intrinsic tactile (IT) 353
 - sensing 460, 466
 - intrinsically safe robot 975, 1338
 - invariance 172
 - inverse
 - dynamics 50, 167, 178, 179, 293, 313
 - dynamics control 141
 - inertia tensor 1046
 - instantaneous kinematics 30
 - kinematics 27, 271, 626, 978
 - kinematics algorithms 254
 - inverse differential kinematics
 - damped least-square solution 252
 - damped least-squares solution with numerical filtering 254
 - least-squares solution 251
 - weighted damped least-squares solution 265
 - involute closure of a distribution 403
 - IPC (interprocess communication) 189, 194, 202
 - IRB-6 967
 - iRobot create 1290
 - ironing robot 1273
 - ISOMAP 1494
 - isotropic configuration 250
 - iterated closest point (ICP) 533
 - iterated extended Kalman filter (IEKF) 97
 - iteratively reweighted least square 99
 - IxTeT 191
-
- ## J
-
- jackknife effect 806
 - Jacobian 29, 45, 59, 136
 - 2-norm condition number 236
 - analytic 248
 - condition number 236
 - dynamically-consistent inverse 45
 - ellipsoid 236
 - extended 257
 - Frobenius norm 236
 - geometric 248
 - matrix 728

- polar decomposition 235
- singular values 235
- task 248
- time-derivative 239
- Jacobian matrix 728
 - generalized 1046
 - pseudoinverse 250
- jamming 657
- Japan Space Exploration Agency (JAXA) 1035
- Japanese experiment module remote manipulator system (JEMRMS) 1035
- Jason 2 988
- Jet Propulsion Laboratory (JPL) 1039, 1042
- joint
 - action 1355
 - attention 1353
 - directors of the laboratories (JDL) 598
 - elasticity 289
 - probability distribution 586
 - reference 1361
 - space 728
 - space control 135
 - torque feedback 301
 - torque sensor 163
- joint architecture for unmanned systems (JAUS) 195
- joint-space
 - inertia matrix (JSIM) 44, 54, 239
 - stiffness matrix 240
- jumping 1405
- just noticeable difference (JND) 721

K

- Kalman filter 96, 590
- Karel 1284
- KARES robots 1238
- Khepera 1288, 1424, 1430, 1435, 1437, 1440, 1445
- kinematic
 - chain topology 230
 - conditioning 724
 - constraint 162, 171
 - duality 659
 - redundancy 245
 - sensors 459
 - singularity 248
 - tree 50
- kinematic calibration 323
 - closed loop 327
 - index 329

- open loop 325
- sensor index 329
- kinematics 727, 728, 975
 - contact 648
 - inverse 27, 271, 626, 978
- kinematics algorithms
 - inverse 254
- kinesthesia 721
- kinesthetic 719, 720
 - displays 731
- kinetic energy of the motors 291
- kinodynamic planning 119
- k -means clustering 100
- knowledge representation (KR) 208
- knowledge-base update 213

L

- L_2 -gain 151
- L1 frequency 486
- labor shortage 1132
- Laboratory for Analysis and Architecture of Systems (LAAS) 947
- LabVIEW 1292
- Lagrange
 - dynamics 134
 - equations 292, 400
 - formulation 44
 - multipliers 172, 366, 400, 729
- L-ALLIANCE 924
- lamprey brain 1416
- landmine 1101, 1103, 1106, 1108, 1109, 1122, 1123
- lane
 - changing 1188
 - keeping 1188
 - tracking 1182
- Langley 1010
- LapTrainer with SimuVision 1216
- LARS surgical robot 1206, 1215
- LaSalle's theorem 139
- laser radar (LADAR) 526, 528
 - or laser detection and ranging 1084
- laser ranging 1043
- laser scanner 92, 968, 1137–1140, 1146, 1147
- laser triangulation 972
- laser-based triangulation 523
- lateral geniculate nucleus (LGN) 1486
- lathe 974
- law 1507
- lead zirconate titanate (PZT) 416
- leader–follower 711
- learning 893, 932, 1396, 1409
 - behavior-based 894, 898, 902
 - by demonstration 902
 - control 155
 - from history 900
 - machine 219
 - reinforcement learning 898, 899, 901
 - statistical 220
- least squares estimation 322
- leg–arm hybrid robots 384
- legged robots 1163
- LEGO 1288
- leg–wheel hybrid robots 383
- Leonardo Da Vinci 989
- Lewellyn's absolute stability criterion 731
- Lie algebra rank condition (LARC) 830
- lift force 993
- light detection and ranging (LIDAR) 523, 528, 1084
- lightweight manipulator 289
- light-weight robot 966
- lightweight unmanned aerial vehicle (UAV) 1013
- likelihood function 586
- Likert 1297
- line of sight (LOS) 1083
- linear
 - constraint satisfaction program (LCSP) 656
 - control design 304
 - program (LP) 683
 - tuning 141
- linearizability
 - feedback 834
- linearization
 - feedback 142, 301, 404
- linearizing coordinates 302
- linkage model
 - virtual 713
- load
 - cell 727
 - distribution 709
 - haul dump vehicle (LDH) 1136
 - haul dump vehicle (LHD) 1130
 - sharing 709
- load-sharing
 - coefficients 709
 - matrix 709
- local and small-time controllability 829
- local steering method 831
- local-area network (LAN) 761

localization
 – sound source 904
 localization (at low cost)
 – absolute 1269
 locally weighted regression (LWR) 1380
 location sensing
 – contact 461
 location-based services 1270
 locomotion 911, 1402
 logic
 – description (DL) 210
 – fuzzy 213
 – propositional 209
 logical sensor system (LSS) 601
 logistics 968
 long-baseline system (LBL) 1000
 longitudinal stability margin 378
 long-term depression 1464
 longwall mine 1141
 loop
 – closure 272
 – haptic 720
 lower pair 19
 Lunokhod 1037
 Lyapunov function 298, 299, 301

M

M+ 931
 m. docilis 1454
 m. speculatrix 1454
 macadamia 1071
 machine
 – learning (ML) 219, 1373
 – tending 968
 – vision 1072, 1432
 machine-tool 280
 machining 974
 magnetic resonance imaging (MRI) 1211, 1212
 magneto-rheological (MR) 1119
 manifesto 1506
 manipulability 725, 1048
 – dynamic 725
 – measure 250
 manipulation
 – admissible configuration 623
 – cooperative 914, 934
 – grasp configuration 624
 – grasp planning 638
 – inverse kinematics 626
 – multiple parts 639
 – multiple robots 639
 – nonprehensile 639
 – pick-and-place task 623

 – planning 122, 622
 – stable part configuration 624
 – task specification 625
 – three DOF manipulator example 625
 – transfer path 623
 – transit path 623
 – workspace goal 628
 manipulation planning
 – hybrid systems 623
 manipulator 1162
 – cooperative 701
 – stiffness 239
 – virtual (VM) 1047
 manned flight 1010, 1012
 manual control 742
 ManuBuild (open building manufacturing) 1082
 manufacturing/assembly 722
 map
 – digital 1180
 – feature 855
 – geometric 855
 – grid 855
 mapping 1146
 – passive 149
 market
 – pull 1279
 – volume 1279
 Markovian decision process (MDP) 218
 Marr–Albus model 1462, 1464
 Mars
 – environmental survey (MESUR) 1041
 – exploration rovers, spirit and opportunity 1043
 – Pathfinder, Sojourner 1041
 – rover sample return (MRSR) 1040
 mass
 – modal 311
 master–slave
 – manipulator (MSM) 1111, 1112, 1118, 1119
 – system 742
 material
 – composition sensing 459
 – transfer 971
 matrix
 – admittance 630
 – dynamic 241
 – exponential 16
 matrix invariance
 – dynamic 241

maximally stable extremal regions (MSER) 554
 maximum a posteriori probability (MAP) 95
 maximum likelihood (ML) 220
 – estimate (MLE) 95
 – estimation (MLE) 951
 – estimator (MLE) 506
 Maxon 727
 McKibben muscle 1466
 mean time between failure (MTBF) 83, 969, 971
 measurement unit (IMU)
 – inertial 482, 995, 1020
 MECANT 1068
 mechanism
 – isotropy 725
 – passive 1116
 – synthesis 271
 mechanoreceptors 734
 mechatronics 1122
 medial temporal area (MT) 1490
 medical
 – image segmentation 1208
 – robots 1200
 – simulation 722
 memory 1396
 MEMS gyroscope 481
 MEMS/MST (microsystem technology) 429
 mental
 – inferences 1365
 – perspective-taking 1362
 mercury project 760
 M-estimator 99
 metal
 – active gas (MAG) 969
 – inert gas (MIG) 969
 metalliferous mining 1127, 1132
 metallo-organic chemical vapor deposition (MOCVD) 422
 metrics 929
 micro and nanoelectromechanical systems (MEMS/NEMS) 411
 micro/nanotechnology 722
 microcomplex 1464
 microelectrical mechanical systems (MEMS) 1123
 microelectromechanical system (MEMS) 412, 498
 microfabrication 419
 microsurgery robots 1213
 microzone 1464
 middleware 194
 milking 1072
 milling machine 974

- mimesis model 1379
- mindreading 1360, 1364
- Mindstorms 1288
- Mine Safety and Health
 - Administration (MSHA) 1159
- Minerals Council of Australia (MCA) 1131
- minimally invasive surgery (MIS) 1204
- minimum mean-square error (MMSE) 95
- mining 1132
 - robotics 1129, 1130
 - vehicle 1127, 1130, 1135, 1143, 1147
- Minorsky 989
- mirror
 - image motion enabler (MIME) 1227, 1230
 - neuron system (MNS) 1363, 1387, 1467, 1469, 1491
 - system 1453
- misconceptions
 - behavior-based 896
- mission control system (MCS) 998
- MIT-MANUS project 1227, 1229
- mixed fleets 1137
- MNS model 1468
- mobile
 - ad hoc network (MANET) 1180
 - arm 968
 - base system (MBS) 1033
 - vocational assistant robot (MoVAR) 1225
- mobile robot
 - nonholonomic 801
- mobility 57
 - degree of 397
 - equation 329
- modal
 - mass 311
 - stiffness 311
 - vectors 241
- mode
 - contact 650
- mode identification and recovery (MIR) 191
- model
 - checking 209
 - contact 647
 - environmental 89
 - predictive control (MPC) 369
 - reference adaptive control (MRAC) 150
- modeling
 - first-order form closure 682
- models and definitions 672
- models of emotions 1356
- modular
 - azimuth positioning system (MAPS) 1143
 - manipulators 915
 - robots 912
- modulation-based range sensors 529
- molecular biology 722
- molecular-beam epitaxy (MBE) 422
- moment labeling 653
- momentum 40
 - conservation 1046
- monitoring
 - environmental 955
- Monte Carlo (MC) 594
- Monterey Bay Aquarium Research Institute (MBARI) 1002
- mood model 1357
- Moon buggy 1037
- Moore–Penrose inverse 143
- moral agents (AMA)
 - artificial 1503
- morality 1505
- morphable model 1492
- morphing
 - motion 1493
- morphological computation 1447
- Mossy fiber (MF) 1464
- motes 1164
- motion
 - control 134, 801, 979
 - effects 513
 - morphing 1493
- motion execution
 - compliant 634
- motion planning
 - compliant 632
 - feedback 635
 - feedback motion planning 635
- motion-oriented operating system (MOOS) 998
- motivated behavioral architecture (MBA) 903
- motivation 898, 903, 904
- motor variables
 - feedback 298
- mouse 720
- MovAid 1238
- move value estimation for robot teams (MVERT) 902
- movement primitive 1494
- moving observation
 - of a corner 514
 - of a plane 513
 - of an edge 515
- moving plate 231
- multi-arm systems
 - cooperative 711
- multi-functional satellite augmentation system (MSAS) 487
- multi-input multi-output (MIMO) 146
- multilateration 485
- multilayer welding 969
- multimodal communication 1350
- multi-mode transit 1194
- multiple operator multiple robot (MOMR) 760
- multiple operator single robot (MOSR) 760
- multiple paired forward-inverse model (MPFIM) 1465
- multiple-view registration 533
- multipulse sonar 511
- multirobot 921, 934
 - path planning 934
 - tasks (MR) 930
- multi-robot systems 891, 896, 897, 899, 902
 - behavior-based 897–899, 902
- multirobot task allocation (MRTA) 930
 - taxonomy 930
- multisensor
 - data fusion 585
 - environment modeling 603
 - fusion architecture 598
 - gripper 1035
 - system control dynamic 603
- multispectral imaging 1138, 1139
- multitarget observation 934
- multitask robots (MT) 930
- MURDOCH 931
- muscle spindles 721
- mutual beliefs 1355

N

- nanoelectromechanical systems (NEMS) 414
- nanorobotic manipulator (NRM) 432
- NASREM (NASA/NBS standard reference model) 192
- National Academy of Engineering 1284
- National Institute of Standards and Technology (NIST) 1165

National Institute on Disability and Rehabilitation Research (NIDRR) 1226
 National Marine Electronics Association (NMEA) 488
 National Space Development Agency of Japan (NASDA) 1035
 natural
 – constraint 174
 – frequency 169, 241, 308
 Naval Postgraduate School (NPS) 992
 NavChair 1238
 navigation
 – inertial (INS) 1143, 1178
 NAVSTAR 485
 NDDS (network data distribution service) 194
 nearness diagram navigation (ND) 843
 Need model 1359
 needle placement robots 1212
 negotiation 930
 Nerd Herd 923
 NeReBot 1231
 network
 – mobility (NEMO) 1180
 – robot forum 945
 – server 759
 – topology 954
 networked robots 943, 1164
 neural network (NN) 220, 1424, 1426–1428, 1433–1435, 1437, 1438, 1441, 1444
 – artificial (ANN) 1373
 – GasNet 1428, 1438, 1439
 – neural oscillator 1427
 – recurrent neural network 1427, 1431, 1432, 1438
 – spiking neural network 1437, 1439
 neurobiological system 1454
 neuroethics 1509
 neuroethology 1453
 neurorobotics 1453
 neutral buoyancy 1036
 Newton–Euler equations 42, 331
 Newton–Raphson 273
 nonaccidental properties (NAPs) 1483
 nonholonomic 1047
 – constraints 828
 – mobile robot 801
 – planning 118
 nonholonomy 403

nonlinear
 – feedback 302
 – optimal control 151
 non-maleficence 1506
 non-minimum-phase 297
 nonprehensile manipulation 647, 666
 nonslip condition 395, 396
 non-uniform rational B-spline (NURBS) 729
 normal vector 729
 Northseeking 989
 Novint
 – Falcon 724
 – Technologies 724
 nuclear
 – decommissioning 1114
 – radiation 1101, 1114
 null-space
 – projection 620
 – velocities 249
 NXT 1291

O

oath 1506
 Oberon 996
 object request broker (ORB) 194
 observability index 337
 observation update 589
 obstacle 540
 – region 110
 – restriction method (ORM) 842
 obstacle avoidance
 – path modification 636
 – problem 838
 occupation health and safety (OH&S) 1085
 octopod 280
 ODIN 1003
 ODIN III 996
 odometry 1398
 offline programming (OLP) 971, 977
 off-read locomotion 1039
 olive 1464
 omni-directional camera 92
 onboard autonomous science investigation system (OASIS) 193
 online
 – gravity compensation 299
 – programming 977
 on-the-fly target classification 511
 ontology 210

open robot controller computer aided design (ORCCAD) 193
 operation
 – interface 1257
 – point 235
 operational
 – amplifier 727
 – space 43
 – space control 135
 operational-space inertia matrix (OSIM) 44, 55
 operator control units (OCU) 195
 optic flow 1437, 1453, 1455
 optical
 – microscope (OM) 433
 – quadrature encoders 725
 optic-flow 1398
 optimal
 – arbitrary time-delay (OAT) 314
 – design 279
 optimization 725
 opto-kinetic response (OKR) 1327
 orbital
 – replacement unit (ORU) 1034, 1036
 – stability 364
 orbiter boom sensor system (OBSS) 1033
 ordinary differential equation (ODE) 992
 ordnance disposal
 – explosive (EOD) 1101, 1103
 orientation
 – absolute 707
 – relative 707
 orthopaedic surgical robot 1209, 1210, 1212
 outer-loop control 142
 OxIM arm 1226

P

P(Y) code 485
 packet 762
 – switching 762
 paddle
 – haptic 726
 painting 971
 – robot 971, 977
 Papert 1284
 paralinguistic cues 1352
 parallel
 – fiber (PF) 1464
 – force/position control 164, 181
 – kinematic machine (PKM) 966, 968, 975

- mechanism 10
- robot 269, 968
- parameter estimation
 - inertial 330
- parameter estimation: links
 - inertial 332
- parameter estimation: loads
 - inertial 330
- parameterized trajectories 369
- parametric bias (PB) 1474
- parking assistance 1187
- Paro seal robot 1227
- partial differential equations (PDE) 309
- partially observable Markovian decision process (POMDP) 218
- passenger protection 1190
- passive
 - action recognition 926
 - dynamic walker 1427
 - mapping 149
 - mechanism 1116
 - walking 363
- passivity 179, 297, 731
- passivity-based control 143
- path
 - deformation 847
 - following 800, 805
 - integration 1398
- Pathfinder 233
- payload delivery 1012, 1013
- PD control 139
- PD force/motion control 712
- pedestrian detection 1184, 1186
- peg-in-hole problem 657
- Peltier 734
- perception 87, 891, 895, 903
 - active 952
 - process 88
- performance 725
- periplaneta computatrix 1455
- perspective taking 1363
- Petri net transducers (PNTs) 191
- phantom 723
- phase 379
- philosophy 1505
- Phoenix 992
- phonotaxis 1399
- phototaxis 1398
- pHRI (physical human–robot interaction) 1335
 - hands-off 1335
 - hands-on 1335
- physical human–robot interaction (pHRI)
 - hands-off 1335
 - hands-on 1335
- physiological proprioception
 - extended (EPP) 1240
- piano mover's problem 111
- pick-and-place operation 978
- PID
 - control 139
 - gain tuning 140
- piezoelectric 498, 734
 - accelerometer 483
- pin in hole 1040
- pinhole camera model 92
- pipe clamp 656
- pitch 989
- place cell 1435, 1436
- placement
 - explosive 1137, 1140, 1147
- plan
 - action 214
 - partial-order 216
 - uncertainty 218
- planar
 - force closure test 687
 - joint 21
 - simplification of contact models 676
- plane, corner and edge moving
 - observation arrival angles 515
- planning 199, 892–894, 904, 1396
 - behavior-based 896, 898, 899, 904
 - closed kinematic chains 639
 - configuration space obstacles 625
 - domain description language (PDDL) 216
 - feedback 634
 - graph 217
 - manipulation 622
 - movable obstacles 639
 - nonholonomic 118
 - robot 219
 - self-collision 626
 - trajectory 120
- PLEXIL (plan execution interchange language) 198
- Plücker coordinates 18, 38
- Plustech 1068
- Pneu-WREX 1231
- Poincaré map 365
- point contact 659
 - without friction (PwoF) 674
- point estimation 94
- point sets 530
- point-light displays 1490
- point-to-point (PTP) 194, 971
- pole placement 303
- policy 218
 - iteration (PI) 218
- POLLY 1292
- polygonal mesh 728
- polygons 729
- polymethyl methacrylate (PMMA) 421
- polytope 239
- polyvinylidene fluoride (PVDF) 416
- POODLE 989
- pool-cleaning robots 1266
- pork 1074
- pose estimation 535
- position
 - absolute 707
 - localization 1178
 - relative 707
- position/force control
 - hybrid 711
- position-based visual servo 572
 - interaction matrix 573
 - stability analysis 574
- positioning
 - absolute 1255
 - accuracy 969
- positive photoresist (PR) 420
- postcondition 215
- posterior inferotemporal cortex (PIT) 1486
- potential
 - damping 992
 - field 1457
 - field method (PFM) 839
- Poultry 1074
- power data grapple fixture (PDGF) 1033
- power-law equation 660, 664, 665
- power-to-weight ratio 726
- precautionary principle 1510
- precise positioning system (PPS) 485
- precondition 215
- predator and prey 1442
- prediction step 589
- predictive graphic display 1035
- prefrontal cortex (PFC) 1486
- pressure
 - distribution 660
 - sensing 462
- preteetum 1457
- preview controller 376
- primal listener 1355

- principal component analysis (PCA) 1380, 1494
- principles
- behavior-based 894, 901
 - sonar 492
- prismatic joint 21
- privacy 1510
- privacy, accuracy, intellectual property, and access (PAPA) 1510
- proactive 955
- probabilistic
- grids 589
 - method 585
 - roadmap method (PRM) 112
- probability distribution
- joint 586
- probability theory 210
- procedural reasoning system (PRS) 198
- procedure call (RPC)
- remote 194
- processing
- tactile 464
- productivity 1130
- professional
- responsibility 1510
 - vocational assistant robot (ProVAR) 1225
- programmable
- logic controller (PLC) 969
 - logic device (PLD) 1445
 - universal machine for assembly (PUMA) 965
- programming by demonstration (PbD) 1371
- project
- ASAP 1413
 - BIOLOCH 1403
 - BISMARC 1414
 - CIRCE 1400
 - Cyber Rodent 1414
 - CYBERHAND 1408
 - ECAGENTS 1414
 - ETHICBOTS 1416
 - ICEA 1401, 1409, 1413
 - MFI 1406
 - Ornithopter 1407
 - RiSE 1405
 - SODAS 1410
 - SWARM-BOTS 1413
- projected gradient method 256
- projection matrix 174, 177
- projective space 125
- property 1510
- proportional derivative (PD) control 296, 298
- proportional–integral–derivative (PID) control 989
- proprioception 87
- proprioceptive sensing 456
- prosody
- affective 1356
- prosthetics 722
- and orthotics (P&O) 1225
- protosocial responses 1356
- proximity sensing 458
- PRS (procedural reasoning system) 191
- pseudo-inertia matrix 136
- pseudoinverse 172
- Jacobian matrix 250
- pseudorandom noise (PRN) 488
- pseudorange 485
- psychophysics 721
- publish–subscribe 188, 194
- pulleys 726
- Puma
- manipulator 1226
 - robot 232
- puncture 732
- pure rolling condition 395, 396
- Purkinje cell (PC) 1463
- pushing 914
- manipulation 658
- pyramidal effect 240
- ## Q
- quadratic optimal control 150
- quality
- transmission 236
- quantization 731
- quantum dot (QD) 414
- quasistatic
- assumption 656
 - equation 677
 - telerobotics (QT) 764
- quaternions 13, 125, 990
- ## R
- R.U.R. (Rossum's Universal Robots) 1308
- radar 1140, 1147
- radial basis function (RBF) 1485
- network 1373
- radiation-induced forces 991
- Radio Technical Commission for Aeronautics (RTCA) 1014
- radiofrequency identification (RFID) 1074, 1084
- based navigation 1270
 - transponder 1269
- Rana computatrix 1455
- random
- loop generator (RLG) 122
 - measure 595
- random-access memory (RAM) 1084
- randomized potential fields 117
- range
- data 521
 - image 521
- range sensor
- time-of-flight (TOF) 528
 - triangulation 529
- ranging
- and wide-bandwidth pulses 506
 - time-of-flight (TOF) 501
- RANSAC feature detection 532
- rapidly adapting lane position handler (RALPH) 1185
- rapidly exploring random tree (RRT) 114, 1321
- ratio
- damping 169
- RC servomotors 734
- reaction null-space (RNS) 1048
- reaction time (RT) 1484
- reactionless manipulation 1048
- reactive
- action package (RAP) 190, 198
 - systems 892–894, 896
- read-only memory (ROM) 1084
- real time 729
- realistic robot simulation (RSS) 971
- reality (VR)
- virtual 1374
- realtime
- control system (RCS) 601
 - kinematics (RTK) 1070
- real-time control system (RCS) 192
- reasoning 891–894, 896, 900, 904
- behavior-based 896, 903
 - symbolic 207
- receiver
- autonomous integrity monitor (RAIM) 488
 - operating curve (ROC) 721
- receptive field weighted regression (RFWR) 1380
- reciprocity 172, 173
- recognition-by-components (RBC) 1483
- recommendation 1506

- reconfigurable modular manipulator
 - system (RMMS) 915
- reconfiguration 916
- reconfigure 945
- reconnaissance 955
- recurrent neural network with
 - parametric bias (RNNPB) 1474
- recursive
 - form of Bayes' rule 586
 - Newton–Euler algorithm (RNEA) 50
- red, green, blue (RGB) 1084
- reduced-order models 708
- redundancy 921
 - kinematic 245
 - resolution via optimization 255
 - resolution via task augmentation 256
- redundant
 - grasp 678
 - manipulator 238
 - robot 238
- reference
 - frame 10
 - joint 1361
 - trajectory generation 153
- referential focus 1361
- reflection
 - force 1111
- refrigerator
 - intelligent 1274
- region of inevitable collision 118
- regional
 - architecture 234
 - structure 231
- registered reflectance image 521
- regolith 1038, 1053
- regressor 147
 - matrix 134
- regulation 135, 1009, 1014
 - control 296
 - of emotion 1356
- regulator 1352
- rehabilitation 722, 1416
 - home-based 1223
- Rehabilitation Engineering Research
 - Center on Rehabilitation Robotics (RERC) 1226
- rehabilitation robotics 1225, 1245
 - safety 1244
- RehaRob 1231
- reinforcement learning (RL) 220, 899, 1386
 - behavior-based 899, 901
- relative
 - orientation 707
 - position 707
- reliability 1511
- remote
 - agent 191
 - center compliance (RCC) 629
 - center of compliance (RCC) 162, 166
 - center of motion (RCM) 1204, 1212
 - control 760
 - handling 1101, 1102, 1112
 - procedure call (RPC) 194
 - sensing 1012
- remotely
 - operated vehicle (ROV) 988, 1103
 - piloted vehicle (RPV) 1014
- REMUS 1004
- rendering 729
 - haptic 720, 727, 735
- rendezvous and docking 1035
- repeatability 260, 969, 970
- representation
 - behavior-based 894, 896, 897, 903
 - singularity 248
- rescue
 - adaptive shoring task 1153
 - beacon or repeater task 1153
 - extrication and evacuation of casualties task 1152
 - in situ medical assessment and intervention task 1152
 - logistics support task 1153
 - reconnaissance and mapping task 1152
 - rubble removal task 1152
 - search task 1152
 - structural inspection task 1152
 - surrogate task 1153
- rescue robot
 - at 2005 hurricane Wilma 1160
 - at La Conchita, California, Mudslide 1158
 - at the 2004 Niigata Chuetsu earthquake 1160
 - at the 2006 Sago Mine disaster 1159
 - at World Trade Center 1157
 - during the 2005 USA hurricane season 1159
 - evaluation 1171
 - modalities 1153
 - sizes 1153
- tasks 1152
- taxonomy 1153
- types of 1153
- resistance temperature device (RTD) 459
- resolution 1506
- responsible conduct of research (RCR) 1521
- restraint analysis 680
- retraction method 115
- revolute joint 20
- Rex 191
- Reynolds number 993
- Rheo Knee 1242
- Riemannian
 - curvature 238
 - manifold 234
- rigid
 - body velocity kinematics 673
 - environment 171, 177, 180
- rigid-body model 45
- rings
 - sonar 512
- risk assessment 975
- road
 - scene understanding 1181
 - sign detection 1182
- Robocow 1074
- RoboCup 1285
 - rescue league 1152
 - rescue simulation project 1165
- Robodoc surgical robot 1205, 1206, 1210
- Roboethics 1504
- robonaut 1057
- robot
 - Aibo 1412
 - anthropopathic 1396
 - architecture 187, 230
 - ASIMO 1399
 - at disasters 1156
 - autonomous 1224
 - bat 1400
 - behavior-based 1408
 - BigDog 1404
 - biohybrid 1396, 1415
 - bio-inspired 1396
 - biomimetic 1396
 - Brachiator 1409
 - Chew Chew 1412
 - Cog 1402, 1411
 - collective 1396
 - component verification on ISS (ROKVISS) 746
 - Conro 1396
 - control 892

- controller 978, 1290
- cricket 1400, 1404
- Cyber Rodent 1414
- Darwin 1409
- design process 229
- EcoBot II 1412
- Elvis 1411
- Entomopter 1414
- gecko 1405
- Golem 1411
- housefly 1398
- HRP2 1399
- humanoid 1399
- hummingbird 1407
- Khepera 1399
- Kismet 1396
- KIV 1410
- language 977
- lobster 1404
- Madeleine 1406
- Mentor 1407
- MFI 1406
- MirrorBot 1409
- octopus 1408
- OCTRAM 1408
- Pathfinder 1414
- planning 219
- platform 1287
- posture 230
- Psikharpx 1409
- reconfigurable 1396
- RoboLobster 1402
- RoboPike 1406
- RoboTuna 1405
- Robur 1411
- Rodolph 1400
- RunBot 1404
- Salamandra Robotica 1402
- s-bot 1413
- Scout II 1404
- SECT 1411
- shape-shifting 1396
- Slugbot 1412
- Spinybot 1405
- Sprawlette 1401
- standards 1167
- Stickybot 1405
- technology experiment (ROTEX) 744, 1035
- test beds 1166
- topology 230
- tournament 1283, 1285
- Tumbleweed 1417
- Uando 1396
- Water Strider 1403
- WE-4RII 1396, 1402

- workcell 963
- worm 1403
- Robota 1239
- robotic
 - arm large and flexible (RALF) 315
 - components verification on the ISS (ROKVISS) 1034
 - work station (RWS) 1034
- roboticized animals 1164
- robotics
 - behavior-based 190, 383
 - health care 1223
 - mining 1129, 1130
 - rehabilitation 1225
- robust
 - estimation methods 98
 - holding 710
- robustness to failures 946
- Roby 2 1003
- rocker bogie chassis 1039
- Rodrigues' equation 16
- roll 989
- rolling contact joint 22
- Romeo 1000, 1003
- room and pillar mine 1141
- Roomba 1290
- rope shovel 1144, 1146
- Rossums Universal Robots 1253
- rotation (CoR)
 - center of 651, 662
- rotation matrix 11
- rotational stiffness 165, 166
- rough-terrain navigation 539
- routing 762
- RTX model 1226
- Rutgers ARM 1231

S

- safety 974, 1201–1208, 1244, 1257
 - category 975
 - standards 1345
- Sagnac effect 480
- saliency map 1461
- salisbury hand 672
- sample based filters 97
- sampling 730
 - and quantization 730
 - importance resampling (SIR) 596
- sampling-based motion planning 111
- satellite 484
- satellite-based augmentation system (SBAS) 487
- saturation 726
- SAUVIM 989, 1002
- s-bot 1443, 1444
- scalability 922, 935, 955
- scan matching 849
- scanning
 - electron microscope (SEM) 417, 418
 - near-field OM (SNOM) 433
 - tunneling microscope (STM) 413, 417, 418
- SCARA standard task 233
- schema 1409
- Schönflies subgroup 233
- Science Applications International, Inc. (SAIC) 1040
- scientific visualization 722
- screw transformation 14
- SE(3) 17
- search and rescue 1012
- section space 265
- secular humanism 1505
- security 1510, 1511
- seek control logic (ASCL)
 - adaptive 1266
- selection matrix 173
- selective
 - availability (SA) 485
 - compliance assembly robot arm (SCARA) 965, 967, 976
 - compliant articulated robot 976
- self-assembly 917
- self-as-simulator 1362
- self-reconfigurable robots 912
- self-repair 918
- self-replicating robots 918
- semiautonomous navigation (SAN) 1040
- semiglobal
 - asymptotic stability (SGAS) 140
 - uniform ultimate boundedness (SGUUB) 140
- semisynchronous orbit 485
- SensAble Technologies 723
- sense-plan-act (SPA) 189
- sensing 87
 - active 634
 - remote 1012
 - vibration 731
- sensor 725, 894
 - classification 90
 - coverage 1256
 - external 1402
 - force 726
 - fusion 1179
 - fusion effects (SFX) 600
 - internal 1402

- kinematic 459
- model 586
- network (ASN), active 602
- tactile 91
- sensor integration
 - tactile 471
- sensor-based control 824
- sensorized environment for life (SELF) 1278
- sensory substitution 732
- separation
 - sound source 904
- sequential Monte Carlo (SMC) 595
- Serafina 1003
- serial
 - chain 10
 - kinematic machine (SKM) 975
- series elastic actuator (SEA) 1317, 1340
- serpentine robots 1162
- SFK meat systems 1074
- shading
 - force 729
- SHAKY 189, 1292
- shape
 - memory alloy (SMA) 427, 734, 1214
- sensing 469
- shared
 - attention 1355, 1361
 - autonomy 748, 1035
 - control 742, 746, 748
- Shear Magic 1073
- shearer position measurement system (SPMS) 1142
- ShearExpress 1073
- sheep shearing 1073
- Sheepdog 1074
- short-baseline system (SBL) 1000
- shuttle remote manipulator system (SRMS) 1032, 1034
- silicon-on-insulator (SOI) 431
- silk-moth antenna 1415
- similarity transformation 242
- simple ranging module rings 512
- simulation and reality 1428, 1434
 - agent–environment relation 1429
 - minimal simulation 1427, 1429
 - physics-based simulation 1427, 1428
- simulation theory 1362
- simultaneous localization and mapping (SLAM) 871, 1001, 1067, 1068
- single
 - operator multiple robot (SOMR) 760
 - operator single robot (SOSR) 760
- single-electron transistors (SET) 438
- single-input single-output (SISO) 1000
- single-robot task (SR) 930
- single-task robot (ST) 930
- singular
 - configuration 248, 724
 - perturbation control 306
 - perturbation model 293
 - region 252
 - value decomposition (SVD) 235, 249, 548, 1048
 - values 235
- singular values 249
- singularity 274
 - avoidance 252
 - dynamic 1047
 - kinematic 248
 - representation 248
 - robustness 251
- sinkage
 - dynamic 1054
- situated robotics 891, 894, 898, 905
- situatedness 891, 892, 894, 895
- situation awareness 1170
- six-DOF robots 279
- skin
 - acceleration sensing 460
 - deflection sensing 463
 - friction 992
- SLAM 1399
- SLAMP 1073
- slaughtering 1073
- slime mold 1416
- slip 732
 - angle 1053
 - displays 733
 - ratio 1053
- smallest singular value 250
- smart
 - appliances 1273
 - floor 1269
 - homes 1275
 - underlay 1269
- smell 1402
- snake robots 1160, 1162
- snake-like robots 262
- SNAME 990
- SO(3) 17
- soccer 935
- sociable robot 1349
- social
 - entropy metric 929
 - referencing 1361
 - robot 1349
 - robot embodiment 1350
- social-cognitive skills 1350
- socially intelligent 1349
- socio-cognitive skills 1360
- soft
 - contact 660
 - finger (SF) 654, 675
- software for distributed robotics (SDR) 951
- Sojourner 233
- solid mechanics 466
- sonar 91, 996
 - beam pattern 494
 - principles 492
- sound source
 - localization 904
 - separation 904
 - tracking 904
- sowing 1071
- space 722
 - joint 728
- space control
 - joint 135
- Space Shuttle 1032
- space station remote manipulator system (SSRMS) 1033, 1034
- Spartacus system 1225
- spatial
 - dynamic voting 767
 - vector notation 37
- special-purpose dexterous manipulator (SPDM) 262, 1034
- specific resistance 386
- specification language for ICE (SLICE) 194
- speech recognition 1119
- spherical
 - joint 21
 - wheel 394
 - wrist 231
- spot welding 966, 970
- spray gun 971
- spraying 1071
- spring 165
 - virtual 728, 729
- square tuning 141
- stability 727
 - margin 378
 - vehicle 1115
- stabilizability 404
- stabilization
 - asymptotic 801

- of fixed points 801
- of trajectories 800
- practical 801, 820
- trajectory 800
- stable
 - contact 730
 - pushing 659
- staged evolution 1426, 1432
- standard 975
 - development kit (SDK) 724
 - end-effector (SEE) 1032
- Stanford arm 966
- Stanford Artificial Intelligence Lab (SAIL) 967
- state
 - affective 1356
- state identification
 - contact 633
- state machine (DSM)
 - dynamic 1445
- state observer 306
- state-displays 1352
- statement 1506
- state-space model 311
- static sinkage 1054
- statics 277
- steady hand robots 1213
- steerability
 - degree of 397
- stereo vision 1043
- Stewart platform 269
- stick
 - virtual 704
- stiffness 165, 277, 728, 974
 - active 166, 167
 - center of 165
 - contact 665
 - control 163, 164, 167
 - matrix 165, 169, 173, 239
 - modal 311
 - translational 165, 166
- stigmergy 925
- stop-and-go 1187
- stopping 1134
- straw-man task 242
- stress rate sensing 460
- strip theory 992
- structural damping 308
- structure and motion 527
- subsumption 189, 1408
 - architecture 600
- summative evaluation 1296
- Sumo 1286
- superior
 - colliculus 1457
 - temporal sulcus (STS) 1469, 1490

- supervisory control 742, 746, 1121
- support
 - pattern 378
 - polygon 373, 378
- surface map 538
- surgeon extender robots 1212
- surgical
 - CAD/CAM 1203, 1204, 1206, 1208–1210, 1212
 - registration 1208
 - robotics 722
 - robots 1215
 - simulation 729
- surgical assistance system 1203, 1204, 1207
- surveillance 955, 1012
- surveyor lunar rover vehicle (SLRV) 1040
- swarm 923, 926, 1164
- swarm-bots 914, 1443, 1444
- Swedish wheel 394, 396
- swimming 1405
- symbol grounding 208
- symmetric formulation 704
- Syritta computatrix 1455
- system 1457
 - behavior-based 891, 893, 896
 - decomposition 188
 - hybrid 893, 894, 896, 898, 902
 - integration 980
 - modeling 89

T

- T3 965
- TA (time-extended assignment) 930
- tactical planning 848
- tactile
 - display 731
 - illusions 735
 - processing 464
 - sensor 91
 - sensor integration 471
- tactile sensing 721
 - dynamic 460, 470
- tactors 732
- tape measure 1083
- TAPs (test action pairs) 193
- target tracking 579
- task
 - allocation 930
 - compatibility 258
 - description 976
 - description language (TDL) 202
 - frame 163, 174
 - Jacobian 248
 - level programming 979
 - oriented kinematics 247
 - priority 258
 - space control 136
 - variable scaling 339
- task description language (TDL) 198
- task space
 - cooperative 707
- task workspace (DTW)
 - dynamic 903
- taste 1402
- taxon 1457
- technical committee (TC) 1504
- technoethics 1504, 1509
- technology push 1279
- tectum 1457
- tegumentum 1457
- tele 741
- tele-actor project 760
- telem Manipulation 742
- teleoperated small emplacement excavator (TSEE) 1113
- teleoperation 720, 742, 759, 1101, 1103–1114, 1117–1119, 1121, 1123, 1135
 - display 1120
- teleoperator 727
- teleorobot 759
- telepresence 742, 1101, 1103
- telerobotic servicer 1032
- telerobotics 1169
- telesensor programming (TSP) 747
- telesurgery 1212
- temperature display
 - haptic 735
- tension-differential type (TDT) 352
- tEODor (telerob explosive ordnance disposal and observation robot) 746
- term-frequency inverse document frequency (TF-IDF) 556
- terrain hazards 1043
- terrain-aided navigation 1000
- Tesla 1010
- tethered walking robots 384
- the effect of a moving observation on the angle of reception 515
- theory of mind 1360
- thermal display 734
- thermal neutron detectors 1106
- thermoelectric coolers 734
- Theseus 1002
- thrusters 997
- Timberjack 1068
- time constraints 214

time update 589
 time-of-flight (TOF) 1269
 – range sensor 528
 – ranging 501
 time-varying feedback 815
 tomography
 – computed (CT) 1211, 1212
 tool
 – virtual 728
 topological localization and mapping 898, 905
 topological map 105
 torque 727
 – computed 303
 torque approach
 – computed 368
 torque control
 – computed 143
 torque feedback
 – joint 301
 torque method
 – computed 802
 torque sensor
 – joint 163
 torsional stiffness 240
 touch 720, 1401
 tour-guide robot 1292
 tournaments 1285
 tracked mechanism 1116, 1117
 tracking
 – sound source 904
 – trajectory 135, 301
 TraderBots 931
 traffic control 934
 traffic-light detection 1183
 trailer systems 812, 823
 trajectory
 – planning 120
 – stabilization 800
 – tracking 135, 301
 – virtual 1494
 transcranial magnetic stimulation (TMS) 1474
 transducer technologies 497
 transfer
 – functions (for an elastic joint) 296
 – matrix method (TMM) 309
 transit 1083
 translational stiffness 165, 166
 transmission 726
 – control protocol (TCP) 762
 – electron microscope (TEM) 418
 – quality 236
 transparent 742
 transportation 1012

transverse function 801, 820
 – approach 821
 tree
 – kinematic 50
 triangulated surfaces 531
 triangulation range sensors 529
 tricept robot 280
 Trinity College's firefighting robot contest (TCFFHRC) 1285
 tripod gait 380
 trot gait 382
 twist 649
 – end-effector 235
 two-handed manipulation 968
 two-point
 – boundary value problem (TPBVP) 260
 – discrimination 733
 two-port network 731
 T-WREX 1231
 types of rescue robots 1153

U

ultra wideband (UWB) 1083
 ultrashort-baseline system (USBL) 1000
 ultrasound (US) imaging 1212
 uncanny valley 1351
 uncertainty 211, 280
 underground coal mining 1140
 underground mining 1127
 – environment 1133
 – ventilation 1133
 underlay
 – smart 1269
 underwater 1101–1103
 – vehicle manipulator system (UVMS) 1001
 unexploded ordnance (UXO) 1101, 1103, 1104, 1107–1109
 uniform ultimate boundedness (UUB) 140
 unit quaternion 166
 universal
 – design principles 1236
 – joint 22
 unmanned
 – aerial system (UAS) 1010
 – aerial vehicle (UAV) 952, 1010, 1012, 1103, 1153, 1163, 1516
 – arial vehicle (UAV) 1075
 – ground vehicle (UGV) 1153
 – surface vehicle (USV) 1153, 1160
 – underwater vehicle (UUV) 995, 1153, 1161
 urban search and rescue 1154
 USAR 1154
 USARSim 1165
 use cases 203
 user data protocol (UDP) 762
 user-centered design 1296
 utilitarianism 1505

V

value iteration (VI) 218
 Van der Pol equation 368
 vapor detector
 – explosive 1107
 vector field histogram (VFH) 841
 vectors
 – modal 241
 vehicle
 – detection 1183
 – mining 1127, 1130, 1135, 1143, 1147
 – stability 1115
 vehicles
 – automated 1191
 vehicular ad-hoc network (VANET) 1180
 velocity
 – analysis 273
 – kinematics 673
 – measurement 726
 – obstacle method (VO) 843
 velocity-resolved control 170, 181
 vertical cell decomposition 115
 vestibular-ocular reflex (VOR) 1327
 vibration 721
 – contact 732
 – display 722
 – feedback 731, 732
 – sensing 731
 – waveforms 732
 vibratory gyroscope 481
 vibrotactile 732
 video game 722
 videoconference 760
 virtual
 – contact manipulator 176
 – coupling 728, 729, 731
 – environment 720, 727, 730
 – fixture 728, 1201, 1206–1208
 – linkage model 713
 – manipulator (VM) 1047
 – reality (VR) 1374
 – spring 728, 729
 – stick 704
 – tool 728

- trajectory 1494
- wall 728, 730, 974
- virtue ethics 1505
- viscoelastic contacts 660
- viscous damping 993
- visibility graph 115
- vision 996, 1398
 - active 1432–1434
- vision systems
 - behavior-based 903
- visual
 - features 564
 - odometry 1056
- visual perspective 1363
- taking 1355
- visual servo control 563
 - 2.5-D 575
 - eye-in-hand systems 580
 - feature trajectory planning 578
 - hybrid approaches 574
 - image-based 565
 - joint space control 580
 - optimization 577
 - partitioned approaches 576
 - position-based 572
 - switching approaches 577
- visually guided behavior 1456
- voice-coil motors 731
- vortex shedding damping 993
- voting-based estimation 99
- voxel 531, 729
- VoxMap pointshell 723
- VVV (versatile volumetric vision) 1316

W

WAAS 486
Wakamaru robot 1239

walker

- haptic 1233

walking 1068, 1403

- passive 363

wall

- virtual 728, 730, 974

wall-climbing 1405

- robot 384

WAMIT 992

wardrobe

- digital 1274

wave

- drift damping 992

waveforms

- vibration 732

waypoint navigation 1039

web 759

webcam 760

webserver 764

wedging 657

weeding 1071

weighted least squares 339

welding 969

- gun 969

well-being products 1235

wheel

- sinkage 1054
- slippage 1038
- traction 1053

wheeled

- mechanism 1115, 1116
- mobile robot (WMR) 391, 799

Wheelseley robots 1238

whiskers 458

whole-arm manipulator (WAM) 1339

wide-area augmentation system (WAAS) 486, 487

wide-area network (WAN) 761

Widrow–Hoff learning rule 1464

window approach (DWA)

- dynamic 842

window-cleaning robots 1267

winner-take-all (WTA) 1456

wire robots 270

work-related musculoskeletal disorders (WMSD) 1336

workspace 25, 276, 725

- dexterous 234
- reachable 234
- topology 234
- volume 234

workspace-shape design rules 233

world graph (WG) 1458

wrench 30, 650

- cone 653
- transmission matrix 332

Wright brothers 1010

wrist center 231

X

XBC 1291
X-ray fluoroscopy 1212

Y

Yamaha 1012
yaw 989

Z

zero dynamics 805, 806
zero-moment point (ZMP) 362, 1494
Z-transform 154